B 9

```
NNN         NNN  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT     AAAAAAAAA        CCCCCCCCCCCC  PPPPPPPPPPPP
NNN         NNN  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT     AAAAAAAAA        CCCCCCCCCCCC  PPPPPPPPPPPP
NNN         NNN  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT     AAAAAAAAA        CCCCCCCCCCCC  PPPPPPPPPPPP
NNN         NNN  EEE                    TTT        AAA         AAA  CCC              PPP         PPP
NNN         NNN  EEE                    TTT        AAA         AAA  CCC              PPP         PPP
NNN         NNN  EEE                    TTT        AAA         AAA  CCC              PPP         PPP
NNNNNN      NNN  EEE                    TTT        AAA         AAA  CCC              PPP         PPP
NNNNNN      NNN  EEE                    TTT        AAA         AAA  CCC              PPP         PPP
NNNNNN      NNN  EEE                    TTT        AAA         AAA  CCC              PPP         PPP
NNN   NNN   NNN  EEEEEEEEEEEE           TTT        AAA         AAA  CCC              PPPPPPPPPPPP
NNN   NNN   NNN  EEEEEEEEEEEE           TTT        AAA         AAA  CCC              PPPPPPPPPPPP
NNN   NNN   NNN  EEEEEEEEEEEE           TTT        AAA         AAA  CCC              PPPPPPPPPPPP
NNN      NNNNNN  EEE                    TTT        AAAAAAAAAAAAAAA  CCC              PPP
NNN      NNNNNN  EEE                    TTT        AAAAAAAAAAAAAAA  CCC              PPP
NNN      NNNNNN  EEE                    TTT        AAAAAAAAAAAAAAA  CCC              PPP
NNN         NNN  EEE                    TTT        AAA         AAA  CCC              PPP
NNN         NNN  EEE                    TTT        AAA         AAA  CCC              PPP
NNN         NNN  EEE                    TTT        AAA         AAA  CCC              PPP
NNN         NNN  EEEEEEEEEEEEEEE        TTT        AAA         AAA     CCCCCCCCCCCC  PPP
NNN         NNN  EEEEEEEEEEEEEEE        TTT        AAA         AAA     CCCCCCCCCCCC  PPP
NNN         NNN  EEEEEEEEEEEEEEE        TTT        AAA         AAA     CCCCCCCCCCCC  PPP
```

```
**FILE**ID**NETDLE                                C 15




NN      NN EEEEEEEEEE TTTTTTTTTT DDDDDDDD   LL         EEEEEEEEEE
NN      NN EEEEEEEEEE TTTTTTTTTT DDDDDDDD   LL         EEEEEEEEEE
NN      NN EE             TT     DD      DD LL         EE
NN      NN EE             TT     DD      DD LL         EE
NNNN    NN EE             TT     DD      DD LL         EE
NNNN    NN EE             TT     DD      DD LL         EE
NN  NN  NN EEEEEEEE       TT     DD      DD LL         EEEEEEEE
NN  NN  NN EEEEEEEE       TT     DD      DD LL         EEEEEEEE
NN    NNNN EE             TT     DD      DD LL         EE
NN    NNNN EE             TT     DD      DD LL         EE
NN      NN EE             TT     DD      DD LL         EE           ....
NN      NN EE             TT     DD      DD LL         EE           ....
NN      NN EEEEEEEEEE     TT     DDDDDDDD   LLLLLLLLLL EEEEEEEEEE   ....
NN      NN EEEEEEEEEE     TT     DDDDDDDD   LLLLLLLLLL EEEEEEEEEE   ....


LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

NE
VO

```
0000            .TITLE  NETDLE - NETACP DLE processing
0000            .IDENT  'V04-000'
0000
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000
0000    ;++
0000    ; FACILITY:     DECnet-VAX
0000    ;
0000    ; ABSTRACT:
0000    ;
0000    ;       This module contains most of the DLE process-level code in
0000    ;       NETACP.  It works with the DLE driver (NDDRIVER) to implement
0000    ;       DLE to allow programs direct access to DECnet circuits.  This
0000    ;       is primarily used to implement MOP support.
0000    ;
0000    ; ENVIRONMENT:
0000    ;
0000    ;       MODE = KERNEL
0000    ;
0000    ; AUTHOR:
0000    ;
0000    ;       Tim Halvorsen, January 1983
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V003    TMH0003         Tim Halvorsen   24-Aug-1984
0000    ;               Prevent duplicate MOM processes from being started due
0000    ;               to unsolicited messages received AFTER MOM has issued
0000    ;               its ACCESS but before it has established a connection
0000    ;               with the node (via SETMODE).  This is done by simply
0000    ;               leaving the unsolicited message which started MOM in the
0000    ;               unsolicited queue for the life of the MOM process, causing
0000    ;               any new unsolicited messages which "squeak through" to be
0000    ;               dropped rather than starting a new MOM process.
0000    ;
0000    ;       V002    TMH0002         Tim Halvorsen   28-Apr-1983
```

```
0000     58 ;                     Change loopback assistant multicast address to the
0000     59 ;                     the one listed in the Ethernet V2 spec.
0000     60 ;
0000     61 ;        V001         TMH0001         Tim Halvorsen   29-Mar-1983
0000     62 ;                     Compute a unique MOM process name, so that multiple
0000     63 ;                     service operations can occur on the same circuit.
0000     64 ;                     Fix deallocation of BC blocks to wait for all outstanding
0000     65 ;                     I/O to rundown before deallocating the block.
0000     66 ;                     Add protective code to prevent multiple MOMs from starting
0000     67 ;                     up if the remote station sends requests too often - if we
0000     68 ;                     receive another message while a MOM process is still starting,
0000     69 ;                     it is ignored.
0000     70 ;--
```

```
                    0000          .SBTTL  Declarations
                    0000  ;
                    0000  ; VMS definitions
                    0000  ;
                    0000
                    0000          $ABDDEF                                 ; ACP buffer descriptor
                    0000          $CCBDEF                                 ; Channel control block
                    0000          $CXBDEF                                 ; Complex buffer
                    0000          $DDBDEF                                 ; Device data block
                    0000          $DDTDEF                                 ; Driver dispatch table
                    0000          $DYNDEF                                 ; Structure types
                    0000          $IRPDEF                                 ; I/O request packet
                    0000          $IODEF                                  ; I/O function codes
                    0000          $JIBDEF                                 ; Job information block
                    0000          $PCBDEF                                 ; Process control block
                    0000          $UCBDEF                                 ; Device unit control block
                    0000
                    0000  ;
                    0000  ; Network definitions
                    0000  ;
                    0000
                    0000          $DWBDEF                         ; DLE window control block
                    0000          $EVCDEF                         ; Event logging parameter codes
                    0000          $LPDDEF                         ; Logical path descriptor (circuit)
                    0000          $NETSYMDEF                      ; Get NET$C_IPL symbol
                    0000          $NFBDEF                         ; Network parameter codes
                    0000          $NMADEF                         ; NICE parameter codes
                    0000          $WQEDEF                         ; Work queue entries
                    0000
                    0000  ;
                    0000  ; Define symbols for timer qualifiers
                    0000  ;
                    0000
          00000001  0000  TID_C_READSUP = 1                       ; NI receive "wait" timer
                    0000
          00000004  0000  WQE$C_QUAL_DLE = 4                      ; && temp &&
                    0000
                    0000  ;
                    0000  ; Define format of broadcast circuit "default protocol user" context block.
                    0000  ; This block holds all context related to enabling this process to receive
                    0000  ; all unsolicited messages ("default user") for the MOP protocol types on
                    0000  ; a broadcast circuit, specifically "load/dump" and "loopback" protocol types.
                    0000  ;
                    0000
                    0000          $DEFINI BC GLOBAL                       ; (GLOBAL is only for debugging)
                    0000
                    0000  $DEF    BC_L_FLINK      .BLKL   2               ; Forward/backward queue links
                    0008  $DEF    BC_W_SIZE       .BLKW   1               ; Size of structure
                    000A  $DEF    BC_B_TYPE       .BLKB   1               ; Type of structure
                    000B  $DEF    BC_B_FLAGS      .BLKB   1               ; Flags
                    000C  _VIELD  BC,0,<-
                    000C          <DELETE,,M>,-                           ; Block is marked for deallocation
                    000C          >
                    000C  $DEF    BC_B_REFCNT     .BLKB   1               ; # of IOWQEs still outstanding
          0000000E  000D                          .BLKB   1               ; (spare for alignment)
                    000E  $DEF    BC_W_LPD        .BLKW   1               ; LPD ID of broadcast circuit
                    0010  $DEF    BC_W_LD_CHAN    .BLKW   1               ; Channel for "load/dump" protocol
```

```
                 0012  $DEF     BC_W_LP_CHAN     .BLKW   1          ; Channel for "loopback" protocol
                 0014  $DEF     BC_Q_PND_RCV     .BLKL   2          ; Listhead of pending receive IOWQEs
                 001C  $DEF     BC_Q_CUR_RCV     .BLKL   2          ; Listhead of current receive IOWQEs
                 0024  $DEF     BC_Q_UNSOL_MSGS  .BLKL   2          ; Listhead for received unsolicited msgs
                 002C  $DEF     BC_C_LENGTH                         ; Length of structure
                 002C
                 002C           $DEFEND BC
                 0000
                 0000  ;
                 0000  ; Define format of an unsolicited message context block
                 0000  ;
                 0000
       0000000E  0000  NIHDRSIZ = 14                                ; Size of NI datalink header
                 0000
                 0000           $DEFINI IOWQE GLOBAL                ; (GLOBAL is only for debugging)
                 0000
       00000C24  0000           . = WQE$C_LENGTH                    ; Start just after standard WQE
                 0024
                 0024  $DEF     IOWQE_Q_IOSB     .BLKL   2          ; I/O status block
       00000026  002C  IOWQE_W_MSGLEN = IOWQE_Q_IOSB+2              ; Message length
                 002C  $DEF     IOWQE_W_CHAN     .BLKW   1          ; Channel to datalink
       00000030  002E                            .BLKW   1          ; (spare for alignment)
                 0030  $DEF     IOWQE_L_PID      .BLKL   1          ; IPID of MOM process for this msg
                 0034  $DEF     IOWQE_L_BC       .BLKL   1          ; Address of corresponding BC block
                 0038  $DEF     IOWQE_G_NIHDR    .BLKB   NIHDRSIZ   ; NI datalink header
                 0046  $DEF     IOWQE_G_MSG      .BLKB   1500       ; Actual message (allow for largest)
                 0622  $DEF     IOWQE_C_LENGTH
                 0622
                 0622           $DEFEND IOWQE
                 0000
                 0000  ;
                 0000  ; Read/write storage
                 0000  ;
                 0000
         00000000                .PSECT  NET_IMPURE,WRT,NOEXE,LONG
                 0000
                 0000  DLE_ACC:
      00000000'  0000           .ADDRESS DLE_ACC                    ; Queue of DLE IO$_ACCESS IRPs
      00000000'  0004           .ADDRESS DLE_ACC                    ; waiting for circuit to go into MOP
                 0008
                 0008  BC_QUEUE:
      00000008'  0008           .ADDRESS BC_QUEUE                   ; Queue of BC blocks for all broadcast
      00000008'  000C           .ADDRESS BC_QUEUE                   ; circuits in the "run" state
                 0010
       00000018  0010  IOSB:    .BLKL   2                           ; General purpose I/O status block
                 0018
         00000000                .PSECT  NET_PURE,NOWRT,NOEXE,LONG
                 0000
                 0000  ;
                 0000  ; Define storage needed to startup MOM
                 0000  ;
                 0000
       0000000A  0000  MAX_MOM_PROC = 10                            ; Maximum number of simultaneous
                 0000                                               ; MOM processes for a single circuit
                 0000  MOM_OBJ_NAM:
4D 4F 4D 24 00'  0000           .ASCIC  "$MOM"                      ; Name of MOM object
             04  0000
```

```
                                    0005  MOM_PRCNAM:
4C 55 21 5F 44 41 21 5F 4D 4F 4D 00'  0005          .ASCIC  "MOM_!AD_!UL"              ; MOM process name
                               0B   0005
                                    0011
                                    0011  ;
                                    0011  ; UNA "setmode" parameters for load/dump protocol
                                    0011  ;
                                    0011
                                    0011  LD_PARAMS:
                         0B0E       0011          .WORD   NMA$C_PCLI_PTY             ; Protocol type = 60-01
                     00000160       0013          .LONG   ^X0160
                         0B1E       0017          .WORD   NMA$C_PCLI_ACC             ; Protocol access mode = SHARED
                     00000001       0019          .LONG   NMA$C_ACC_SHR
                         0AF1       001D          .WORD   NMA$C_PCLI_BUS             ; Buffer size = 1498 (2 bytes for PAD)
                     000005DA       001F          .LONG   1498
                         0451       0023          .WORD   NMA$C_PCLI_BFN             ; Number of buffers = 2
                     00000002       0025          .LONG   2
                         0B0F       0029          .WORD   NMA$C_PCLI_MCA             ; Reception of multicast messages:
                         0008       002B          .WORD   8                          ;       (8 byte string follows)
                         0001       002D          .WORD   NMA$C_LINMC_SET            ; Enable reception of multicast
                     010000AB       002F          .LONG   ^X010000AB                 ;    "dump/load assistance"
                         0000       0033          .WORD   0
                         0B1A       0035          .WORD   NMA$C_PCLI_PAD             ; Padding length word = ON
                     00000000       0037          .LONG   NMA$C_STATE_ON
                         0B18       003B          .WORD   NMA$C_PCLI_PRM             ; Promiscuous mode = OFF
                     00000001       003D          .LONG   NMA$C_STATE_OFF
                         0B19       0041          .WORD   NMA$C_PCLI_MLT             ; Multicast address state = OFF
                     00000001       0043          .LONG   NMA$C_STATE_OFF
                         0B1B       0047          .WORD   NMA$C_PCLI_DCH             ; Data chaining = OFF
                     00000001       0049          .LONG   NMA$C_STATE_OFF            ; (DLE driver can't handle multiple CXBs)
                         0B1C       004D          .WORD   NMA$C_PCLI_CRC             ; CRC generation = ON
                     00000000       004F          .LONG   NMA$C_STATE_ON
                                    0053
                                    0053  LD_SETMODE:
                     00000042       0053          .LONG   .-LD_PARAMS                ; Descriptor of above buffer
                    00000011'       0057          .ADDRESS LD_PARAMS
                                    005B
                                    005B  ;
                                    005B  ; UNA "setmode" parameters for loopback protocol
                                    005B  ;
                                    005B
                                    005B  LP_PARAMS:
                         0B0E       005B          .WORD   NMA$C_PCLI_PTY             ; Protocol type = 90-00
                     00000090       005D          .LONG   ^X0090
                         0B1E       0061          .WORD   NMA$C_PCLI_ACC             ; Protocol access mode = SHARED
                     00000001       0063          .LONG   NMA$C_ACC_SHR
                         0AF1       0067          .WORD   NMA$C_PCLI_BUS             ; Buffer size = 1500
                     000005DC       0069          .LONG   1500
                         0451       006D          .WORD   NMA$C_PCLI_BFN             ; Number of buffers = 2
                     00000002       006F          .LONG   2
                         0B0F       0073          .WORD   NMA$C_PCLI_MCA             ; Reception of multicast messages:
                         0008       0075          .WORD   8                          ;       (8 byte string follows)
                         0001       0077          .WORD   NMA$C_LINMC_SET            ; Enable reception of multicast
                     000000CF       0079          .LONG   ^X000000CF                 ;    "loopback assistance"
                         0000       007D          .WORD   0
                         0B1A       007F          .WORD   NMA$C_PCLI_PAD             ; Padding length word = OFF
                     00000001       0081          .LONG   NMA$C_STATE_OFF
```

```
            0B18  0085            .WORD   NMA$C_PCLI_PRM          ; Promiscuous mode = OFF
        00000001  0087            .LONG   NMA$C_STATE_OFF
            0B19  008B            .WORD   NMA$C_PCLI_MLT          ; Multicast address state = OFF
        00000001  008D            .LONG   NMA$C_STATE_OFF
            0B1B  0091            .WORD   NMA$C_PCLI_DCH          ; Data chaining = OFF
        00000001  0093            .LONG   NMA$C_STATE_OFF         ; (DLE driver can't handle multiple CXBs)
            0B1C  0097            .WORD   NMA$C_PCLI_CRC          ; CRC generation = ON
        00000000  0099            .LONG   NMA$C_STATE_ON
                  009D
                  009D    LP_SETMODE:
        00000042  009D            .LONG   .-LP_PARAMS             ; Descriptor of above buffer
       0000005B'  00A1            .ADDRESS LP_PARAMS
                  00A5
                  00A5
          00000000                .PSECT  NET_CODE,NOWRT,EXE
```

```
                        0000          .SBTTL  DLE$DISPATCH - Dispatch newly recieved DLE IRP
                        0000  ;+
                        0000  ; DLE$DISPATCH - Dispatch newly received DLE IRP
                        0000  ;
                        0000  ; This routine is called from AQB dispatching when an IRP is dequeued
                        0000  ; which has the PHYSIO flag set in the IRP flags.  This flag is used
                        0000  ; by convention between NETDRIVER and NDDRIVER to distinguish between
                        0000  ; various flavors of IRPs.
                        0000  ;
                        0000  ; Inputs:
                        0000  ;
                        0000  ;       R3 = IRP address
                        0000  ;
                        0000  ; Outputs:
                        0000  ;
                        0000  ;       None - the IRP is always returned to the driver.
                        0000  ;-
                        0000  DLE$DISPATCH::
              00   EF   0000          EXTZV   #IRP$V_FCODE,-          ; Get function code
              06        0002                  #IRP$S_FCODE,-
        57  20 A3       0003                  IRP$W_FUNC(R3),R7
                        0006          $DISPATCH  R7,<-
                        0006                  <IO$_ACCESS,     30$>,-
                        0006                  <IO$_ACPCONTROL, 40$>,-
                        0006                  <IO$_DEACCESS,   50$>,-
                        0006                  <IO$_SETMODE,    60$>>
       0000'8F     3C   0036  10$:    MOVZWL  #SS$_ILLIOFUNC,-        ; Say "illegal I/O function"
           38 A3        003A                  IRP$L_IOST1(R3)
              28   11   003C          BRB     90$                     ; Exit
                        003E          ;
                        003E          ;   ACCESS function - dispatch to connect processor
                        003E          ;
            0034   30   003E  30$:    BSBW    DLE$ACCESS              ; Process IO$_ACCESS function
              23   11   0041          BRB     90$                     ; Exit
                        0043          ;
                        0043          ;   ACP Control
                        0043          ;
              03   E0   0043  40$:    BBS     #IRP$V_COMPLX,-         ; If normal IO$_ACPCONTROL, then
        EE 2A A3        0045                  IRP$W_STS(R3),10$       ; inform user we don't support them
     18 A3    01   8A   0048          BICB    #1,IRP$L_WIND(R3)       ; Clear interlock bit in case an
                        004C                                          ; IO$_ACCESS or IO$_DEACCESS is pending
            02E2   30   004C          BSBW    DLE$CANCEL              ; Do cancel-related work
              15   11   004F          BRB     90$                     ; Continue
                        0051          ;
                        0051          ;   DEACCESS function
                        0051          ;
  56  18 A3   01   CB   0051  50$:    BICL3   #1,IRP$L_WIND(R3),R6    ; Get DWB without interlock bit
              0E   18   0056          BGEQ    90$                     ; If GEQ then no DWB
            024C   30   0058          BSBW    DLE$DEACCESS            ; Process IO$_DEACCESS function
              09   11   005B          BRB     90$                     ; Continue
                        005D          ;
                        005D          ;   SETMODE function
                        005D          ;
      56  18 A3    D0   005D  60$:    MOVL    IRP$L_WIND(R3),R6       ; Get DWB address
              03   18   0061          BGEQ    90$                     ; If GEQ then no DWB
            01B9   30   0063          BSBW    DLE$SETMODE             ; Process IO$_SETMODE function
                        0066          ;
```

```
                              0066   314             ;   Give the IRP back to the DLE driver with the I/O status setup
                              0066   315             ;
                   53    D5   0066   316 90$:        TSTL    R3                          ; Did IRP get tucked away somewhere
                   0A    13   0068   317             BEQL    100$                        ; If so, exit
            55   1C  A3    D0   006A   318             MOVL    IRP$L_UCB(R3),R5            ; Get UCB address
          00000000'GF    16   006E   319             JSB     G^EXE$INSIOQ                ; Queue packet to driver
                         05   0074   320 100$:       RSB                                 ; Done
```

```
                                 0075          .SBTTL  DLE$ACCESS      - Handle IO$_ACCESS function
                                 0075   ;+
                                 0075   ; DLE$ACCESS - Process IO$_ACCESS function for a DLE channel
                                 0075   ;
                                 0075   ; This routine is entered after the initial IO$_ACCESS processing
                                 0075   ; done in the DLE driver.  It's main function is to perform all
                                 0075   ; those things which must be done in process context in order to
                                 0075   ; setup the connection between DLE user and the datalink.
                                 0075   ;
                                 0075   ; Inputs:
                                 0075   ;
                                 0075   ;       R3 = IRP address
                                 0075   ;
                                 0075   ;       P1 = Circuit name for DLE I/O
                                 0075   ;
                                 0075   ; Outputs:
                                 0075   ;
                                 0075   ;       R3 = IRP address, 0 if not to be returned to driver yet.
                                 0075   ;       IRP$L_IOST1 = I/O status
                                 0075   ;-
                                 0075   DLE$ACCESS:
            54 A3      D4        0075          CLRL    IRP$L_EXTEND(R3)        ; Assume no rcvd msg to be returned
                                 0078          ;
                                 0078          ;   Construct a descriptor of the circuit name
                                 0078          ;
            2C B3      C1        0078          ADDL3   @IRP$L_SVAPTE(R3),-     ; Get address of P1 ABD
         54    08                007B                  #ABD$C_FIB*ABD$C_LENGTH,R4
      57   02 A4      3C         007D          MOVZWL  ABD$W_COUNT(R4),R7      ; Get length of circuit name
         51    64      3C        0081          MOVZWL  ABD$W_TEXT(R4),R1       ; Get offset to circuit name
   58    01 A441      9E         0084          MOVAB   1+ABD$W_TEXT(R4)[R1],R8 ; Get address of text (skip acmode)
                                 0089          ;
                                 0089          ;   Locate the CRI and LPD for the circuit, and make sure it is
                                 0089          ;   in a state to handle MOP mode.
                                 0089          ;
 5B    00000000'EF     D0        0089          MOVL    NET$GL_CNR_CRI,R11      ; Point to CRI root block
               5A      D4        0090          CLRL    R10                     ; Start at beginning of CRI list
   50     0000'8F      3C        0092          MOVZWL  #SS$_NOSUCHDEV,R0       ; Setup default error code
                                 0097          $SEARCH eql,cri,s,nam           ; Lookup CRI by circuit name
            48 50      E9        00A6          BLBC    R0,91$                  ; If error detected, then report it
                                 00A9          $GETFLD cri,l,sta               ; Get circuit state
   50     0000'8F      3C        00B6          MOVZWL  #SS$_DEVINACT,R0        ; Assume circuit not on
       01      58      D1        00BB          CMPL    R8,#NMA$C_STATE_OFF     ; Circuit off?
               31      13        00BE          BEQL    91$                     ; If so, report an error
            FF3D'      30        00C0          BSBW    NET$LOCATE_LPD          ; Get LPD address
            2B 50      E9        00C3          BLBC    R0,91$                  ; Exit if error detected
     50     4C A3      D0        00C6          MOVL    IRP$L_DIAGBUF(R3),R0    ; Get DWB address
            20 A6      B0        00CA          MOVW    LPD$W_PTH(R6),-         ; Store LPD ID of circuit
            3E A0                00CD                  DWB$W_PATH(R0)          ; into DLE window block
                                 00CF          $GETFLD cri,v,ser               ; Service functions enabled?
   50     0000'8F      3C        00DC          MOVZWL  #SS$_IVMODE,R0          ; Assume service disabled
            0D 58      E8        00E1          BLBS    R8,91$                  ; If disabled, then report error
               07      E0        00E4          BBS     #LPD$V_X25,-            ; No service is allowed
         08 22 A6                00E6                  LPD$W_STS(R6),91$       ; on X.25 DLM circuits
                                 00E9          ;
                                 00E9          ;   If this is a multiaccess circuit, such as Ethernet,
                                 00E9          ;   then skip the circuit transition, since there is no
                                 00E9          ;   circuit "mode".
```

```
                              00E9            ;
                  0A   E1     00E9            BBC     #LPD$V_BC,-             ; Skip if not broadcast
            06 22 A6          00EB                    LPD$W_STS(R6),10$
                00CF   30     00EE            BSBW    BC_ACCESS               ; Handle broadcast DLE access
                005E   31     00F1      91$:  BRW     90$                     ; Return status to DLE driver
                              00F4      10$:  ;
                              00F4            ;   Mark the DLE process as the owner of the circuit.  If the
                              00F4            ;   circuit is already owned, return an error.
                              00F4            ;
                              00F4            $GETFLD cri,l,owpid             ; Get PID of DLE owner
               0D 50   E9     0101            BLBC    R0,20$                  ; Branch if not currently owned
           0C A3  58   D1     0104            CMPL    R8,IRP$L_PID(R3)        ; Is it already owned by process?
                  07   13     0108            BEQL    20$                     ; If so, ok to access
          50   0000'8F 3C     010A      15$:  MOVZWL  #SS$_DEVALLOC,R0        ; Report circuit already owned
                  41   11     010F            BRB     90$
                  03   E2     0111      20$:  BBSS    #LPD$V_ACCESS,-         ; Mark circuit accessed for DLE
            F4 22 A6          0113                    LPD$W_STS(R6),15$       ; If already accessed, report error
          58   0C A3   D0     0116            MOVL    IRP$L_PID(R3),R8        ; Get caller's PID
               FEE3'   30     011A            BSBW    CNF$PUT_FIELD           ; Make process owner of the circuit
                  02   E2     011D            BBSS    #LPD$V_DLE,-            ; Mark in DLE mode
            1D 22 A6          011F                    LPD$W_STS(R6),30$       ; If already in DLE, skip logging event
                              0122            ;
                              0122            ;   Log an event indicating the circuit has been accessed
                              0122            ;   locally by a process.
                              0122            ;
     55   00000000'EF  9E     0122            MOVAB   NET$AB_EVT_WQE,R5       ; Get address of common WQE
               20 A6   B0     0129            MOVW    LPD$W_PTH(R6),-         ; Set LPD ID into WQE
               12 A5          012C                    WQE$W_REQIDT(R5)
             0140 8F   B0     012E            MOVW    #EVC$C_DLL_LSC,-        ; "locally initiated state change"
               1C A5          0132                    WQE$W_EVL_CODE(R5)
                  03   90     0134            MOVB    #EVC$C_DLL_POLD_RUNG,-  ; Old state = RUNNING
               1E A5          0136                    WQE$B_EVL_DT1(R5)
                  04   90     0138            MOVB    #EVC$C_DLL_POLD_MAIN,-  ; New state = MAINTAINANCE
               1F A5          013A                    WQE$B_EVL_DT2(R5)
               FEC1'   30     013C            BSBW    NET$EVT_INTRAW          ; Log the event record
                              013F            ;
                              013F            ;   Bring the circuit up in "MOP" state.
                              013F            ;
          50   0000'8F 3C     013F      30$:  MOVZWL  #LEV$C_DLE_ACC,R0       ; Setup DLLTRN event code
               FEB9'   30     0144            BSBW    SET_DLL_EVT             ; Queue the request
                              0147            ;
                              0147            ;   Wait for the circuit to become ready.  When it does, the
                              0147            ;   routine DLE$LPD_STATUS will be called.
                              0147            ;
    00000004'FF   63   0E     0147            INSQUE  (R3),@DLE_ACC+4         ; Insert IRP onto waiting queue
                  53   D4     014E            CLRL    R3                      ; Indicate IRP not to be returned
                  04   11     0150            BRB     100$
                              0152
                              0152      ;
                              0152      ; An error has been detected.  Return the IRP back to the driver.
                              0152      ;
                              0152
           38 A3  50   3C     0152      90$:  MOVZWL  R0,IRP$L_IOST1(R3)      ; Pass status back in IRP
                       05     0156      100$: RSB
```

```
                               0157          .SBTTL  DLE$LPD_STATUS - Check completion of MOP transition
                               0157     ;+
                               0157     ; DLE$LPD_STATUS - Check completion of MOP transition
                               0157     ;
                               0157     ; This routine is called when an LPD has made the transition into MOP
                               0157     ; state or if an error has occurred.  It is always called by DLLTRN
                               0157     ; on circuit transitions if the ACCESS flag is set in the LPD.
                               0157     ;
                               0157     ; If there is a process waiting to access the circuit, then if the
                               0157     ; transition was successful, then that process is allowed to proceed
                               0157     ; with the access.
                               0157     ;
                               0157     ; Inputs:
                               0157     ;
                               0157     ;       R6 = LPD address
                               0157     ;       R0 = Status of attempted MOP transition of circuit
                               0157     ;
                               0157     ; Outputs:
                               0157     ;
                               0157     ;       None
                               0157     ;
                               0157     ;       R0-R3,R8-R9 are destroyed.
                               0157     ;-
                               0157     DLE$LPD_STATUS::
                    30    BB   0157             PUSHR   #^M<R4,R5>              ; Save registers
                               0159             ;
                               0159             ;   Locate the DWB corresponding to the process attempting
                               0159             ;   the circuit ACCESS.
                               0159             ;
     51   00000000'EF    9E    0159             MOVAB   DLE_ACC,R1              ; Get address of DLE ACCESS IRP listhead
                 53   51    D0   0160             MOVL    R1,R3                   ; Setup for loop
                 53   63    D0   0163     10$:    MOVL    (R3),R3                 ; Skip to next IRP in list
                 51   53    D1   0166             CMPL    R3,R1                   ; End of list?
                      39    13   0169             BEQL    60$                     ; If so, then ignore the status
           54    4C   A3    D0   016B             MOVL    IRP$L_DIAGBUF(R3),R4    ; Get DWB address for ACCESS request
                 20   A6    B1   016F             CMPW    LPD$W_PTH(R6),-         ; Is it for this circuit?
                 3E   A4         0172                     DWB$W_PATH(R4)
                      ED    12   0174             BNEQ    10$                     ; If not, keep looking
                 53   63    0F   0176             REMQUE  (R3),R3                 ; Remove from pending ACCESS list
                 10   50    E9   0179             BLBC    R0,20$                  ; Branch if circuit is down
                                 017C             ;
                                 017C             ;   Setup the datalink channel and UCB address in DWB
                                 017C             ;
                 14   A6    B0   017C             MOVW    LPD$W_CHAN(R6),-        ; Save channel to datalink
                 4C   A4         017F                     DWB$W_DLL_CHAN(R4)
                 10   A6    D0   0181             MOVL    LPD$L_UCB(R6),-         ; Save UCB of datalink
                 48   A4         0184                     DWB$L_DLL_UCB(R4)
                                 0186             ;
                                 0186             ;   Set the circuit substate to "auto-service"
                                 0186             ;
                      06    90   0186             MOVB    #NMA$C_LINSS_ASE,-      ; Set circuit substate
                 27   A6         0188                     LPD$B_SUB_STA(R6)
                      08    11   018A             BRB     50$                     ; Pass success back to driver
                                 018C             ;
                                 018C             ;   Failure to make transition - reset LPD to original state
                                 018C             ;
                      50    DD   018C     20$:    PUSHL   R0                      ; Save final status
```

```
             0167   30  018E         BSBW    LEAVE_MOP_STATE          ; Leave MOP state
               50 8ED0  0191         POPL    R0                       ; Restore final status
                        0194         ;
                        0194         ;   Report the status back to DLE driver
                        0194         ;
        38 A3    50  B0  0194  50$:   MOVW    R0,IRP$L_IOST1(R3)       ; Store status in IRP
        55    1C A3  D0  0198         MOVL    IRP$L_UCB(R3),R5         ; Point to the DLE UCB
       00000000'GF   16  019C         JSB     G^EXE$INSIOQ             ; Queue packet to DLE driver
                19   11  01A2         BRB     90$
                         01A4
                         01A4  60$:   ;
                         01A4         ;   There is no ACCESS request pending for this circuit.  If
                         01A4         ;   the LPD status is "success", then we can ignore it, since
                         01A4         ;   its not relevant except to restart a pending ACCESS.
                         01A4         ;
             16 50   E8  01A4         BLBS    R0,90$                   ; Exit if LPD is ok
                         01A7         ;
                         01A7         ;   There may be an active DLE session currently in progress
                         01A7         ;   over this circuit.  Tell the DLE driver to locate all DWBs
                         01A7         ;   associated with this circuit, and if any, to abort them.
                         01A7         ;
        58    20 A6  3C  01A7         MOVZWL  LPD$W_PTH(R6),R8         ; Pass path ID to driver
  55   00000000'EF   D0  01AB         MOVL    NET$GL_DLE_UCB,R5        ; Get DLE UCB address
     51    0088 C5   D0  01B2         MOVL    UCB$L_DDT(R5),R1         ; Get DDT address
           04 B1     16  01B7         JSB     @DDT$L_UNSOLINT(R1)      ; Call "LPD down" entry point
                         01BA                                          ; with R0 = status code
                         01BA                                          ;  and R8 = path ID
                         01BA         ;
                         01BA         ;   Leave MOP state
                         01BA         ;
             013B   30  01BA         BSBW    LEAVE_MOP_STATE          ; Leave MOP state
               30   BA  01BD  90$:   POPR    #^M<R4,R5>               ; Restore registers
                    05  01BF         RSB
```

```
                         01C0          .SBTTL  BC_ACCESS - Handle DLE access to broadcast circuit
                         01C0      ;+
                         01C0      ; BC_ACCESS - Handle DLE access to multiaccess circuit
                         01C0      ;
                         01C0      ; This routine is called when an access is being attempted to an
                         01C0      ; Ethernet.  Since there is no "MOP mode" for multiaccess circuits,
                         01C0      ; we simply assign a new channel to the device, issue a SETMODE to
                         01C0      ; enable access to a given destination, and complete the access.
                         01C0      ;
                         01C0      ; Inputs:
                         01C0      ;
                         01C0      ;       R3 = IRP address for ACCESS request
                         01C0      ;       R6 = LPD address
                         01C0      ;       R10/R11 = CNF/CNR addresses for CRI
                         01C0      ;
                         01C0      ; Outputs:
                         01C0      ;
                         01C0      ;       R0 = Status code
                         01C0      ;-
                         01C0  BC_ACCESS:
                         01C0              ;
                         01C0              ;    Make sure the circuit is in the "run" state
                         01C0              ;
             04    E0    01C0              BBS     #LPD$V_RUN,-                ; If circuit not ready,
       08 22 A6          01C2                      LPD$W_STS(R6),10$
   50   0000'8F    3C    01C5              MOVZWL  #SS$_DEVINACT,R0            ; Return "circuit not on"
           0051    31    01CA              BRW     90$                         ; Report the error
                         01CD  10$:        ;
                         01CD              ;    Set a flag in the DWB indicating that this is an NI.
                         01CD              ;
    54    4C A3    D0    01CD              MOVL    IRP$L_DIAGBUF(R3),R4        ; Get DWB address
                         01D1              SETBIT  #DWB$V_BC,DWB$W_FLAGS(R4)   ; Indicate circuit is an NI
                         01D6              ;
                         01D6              ;    Assign a new channel for this DLE session.  Each DLE
                         01D6              ;    session uses a new NETACP channel so that the demultiplexing
                         01D6              ;    done by the datalink for received messages (based on the
                         01D6              ;    source node) can be used by the DLE driver to distinguish
                         01D6              ;    incoming messages between the various DLE users.
                         01D6              ;
   50   0000'8F    3C    01D6              MOVZWL  #SS$_NOSUCHDEV,R0           ; Setup default error code
                         01DB              $GETFLD cri,s,vmsnam                ; Get datalink device name
          33 50    E9    01E8              BLBC    R0,90$                      ; Exit if error detected
        7E   57    7D    01EB              MOVQ    R7,-(SP)                    ; Push descriptor on stack
        50   5E    D0    01EE              MOVL    SP,R0                       ; Get address of descriptor
                         01F1              $ASSIGN_S DEVNAM=(R0),-             ; Assign a new channel for DLE
                         01F1                      CHAN=DWB$W_DLL_CHAN(R4)
        5E   08    C0    01FF              ADDL    #8,SP                       ; Pop descriptor off stack
          19 50    E9    0202              BLBC    R0,90$                      ; Exit if error detected
             53    DD    0205              PUSHL   R3                          ; Save IRP address
    50    4C A4    3C    0207              MOVZWL  DWB$W_DLL_CHAN(R4),R0       ; Get channel number
     00000000'GF   16    020B              JSB     G^IOC$VERIFYCHAN            ; Get the CCB address; ignore errors
             53  8ED0    0211              POPL    R3                          ; Restore IRP address
             61    D0    0214              MOVL    CCB$L_UCB(R1),-             ; Save the datalink UCB address
          48 A4          0216                      DWB$L_DLL_UCB(R4)
           04E5    30    0218              BSBW    ATTACH_UNSOL_MSG            ; Pass unsolicited message to user
        50   00'   D0    021B              MOVL    S^#SS$_NORMAL,R0            ; Success
             05          021E  90$:        RSB                                 ; Exit with status
```

```
                       021F          .SBTTL  DLE$SETMODE - Process IO$_SETMODE request
                       021F  ;+
                       021F  ; DLE$SETMODE - Process IO$_SETMODE request at process level
                       021F  ;
                       021F  ; This routine is called to perform all work needed for the DLE SETMODE
                       021F  ; QIO at IPL 0.  This includes issuing a SETMODE function to the datalink
                       021F  ; driver on the DLE user's behalf.  Most of the wo-k done for the SETMODE
                       021F  ; has already been accomplished by the DLE driver.
                       021F  ;
                       021F  ; Inputs:
                       021F  ;
                       021F  ;       R6 = DWB address
                       021F  ;       R3 = IRP address
                       021F  ;
                       021F  ;       P2 = UNA P2 buffer (used only for DLE access to UNA)
                       021F  ;       P3 = Ethernet remote address (used only for DLE access to UNA)
                       021F  ;       P4 = Substate
                       021F  ;
                       021F  ; Outputs:
                       021F  ;
                       021F  ;       R3 = IRP address, 0 if not to be returned to driver yet.
                       021F  ;       IRP$L_IOST1 = I/O status
                       021F  ;-
                       021F  DLE$SETMODE:
                       021F          ;
                       021F          ;   For point-to-point circuits, propagate the (possibly) updated
                       021F          ;   circuit substate to the LPD (it has already been set in the
                       021F          ;   DWB by the driver) so that we can see it with existing network
                       021F          ;   management.
                       021F          ;
          03     E0    021F          BBS     #DWB$V_BC,-             ; If point-to-point circuit.
     17 0E A6          0221                  DWB$W_FLAGS(R6),10$
  58    3E A6    3C    0224          MOVZWL  DWB$W_PATH(R6),R8       ; Get LPD ID
          56     DD    0228          PUSHL   R6                      ; Save DWB address
       FDD3'     30    022A          BSBW    NET$FIND_LPD            ; Locate LPD
     52   56     D0    022D          MOVL    R6,R2                   ; Set LPD address in R2
          56   8ED0    0230          POPL    R6                      ; Restore DWB address
       05 50     E9    0233          BLBC    R0,10$                  ; If cannot be found, skip it
       46 A6     90    0236          MOVB    DWB$B_SUBSTA(R6),-      ; Copy substate value to LPD
       27 A2           0239                  LPD$B_SUB_STA(R2)
                       023B  10$:    ;
                       023B          ;   Construct a descriptor of the P2 buffer (UNA P2 buffer).
                       023B          ;   If none specified, then skip the SETMODE.
                       023B          ;
       2C B3     C1    023B          ADDL3   @IRP$L_SVAPTE(R3),-     ; Get address of P2 ABD
     54   10           023E                  #ABD$C_NAME*ABD$C_LENGTH,R4
  57    02 A4    3C    0240          MOVZWL  ABD$W_COUNT(R4),R7      ; Get length of P2
          59     13    0244          BEQL    40$                     ; Skip if none
     51   64     3C    0246          MOVZWL  ABD$W_TEXT(R4),R1       ; Get offset to P2 data
  58    ^1 A441  9E    0249          MOVAB   1+ABD$W_TEXT(R4)[R1],R8 ; Get address of P2 data (skip acmode)
                       024E          ;
                       024E          ;   Issue a SETMODE to the datalink driver to establish
                       024E          ;   "shared" access to the remote node.  This allows
                       024E          ;   more than one DLE user to use the protocol type at the
                       024E          ;   same time - demultiplexing is done for received messages
                       024E          ;   based on the remote node address.
                       024E          ;
```

```
                7E   57   7D  024E   640          MOVQ    R7,-(SP)                  ; Push descriptor of UNA P2 buffer
                50   5E   D0  0251   641          MOVL    SP,R0                     ; Get address of descriptor
      52   00000010'EF   9E  0254   642          MOVAB   IOSB,R2                   ; Get address of I/O status block
                              025B   643          $QIOW_S FUNC=#IO$_SETMODE!IO$M_CTRL!IO$M_STARTUP,- ; Issue request
                              025B   644                  CHAN=DWB$W_DLL_CHAN(R6),-
                              025B   645                  EFN=#NET$C_EFN_WAIT,-
                              025B   646                  IOSB=(R2),-
                              025B   647                  P2=R0
                5E   08   C0  0279   648          ADDL    #8,SP                     ; Pop descriptor off stack
             23   50   E9  027C   649          BLBC    R0,90$                    ; Exit if error detected
             50   62   3C  027F   650          MOVZWL  (R2),R0                   ; Get final I/O status
             07   50   E8  0282   651          BLBS    R0,30$                    ; Exit if ok
        3C A3   04 A2   D0  0285   652          MOVL    4(R2),IRP$L_IOST2(R3)     ; Return UNA longword to user
                  16   11  028A   653          BRB     90$                       ; Store primary status and exit
                              028C   654 30$:     ;
                              028C   655          ;    As a result of a SETMODE to the UNA driver for LIMITED protocol
                              028C   656          ;    access, the UNA driver may have evaporated the UCB we initially
                              028C   657          ;    got after the $ASSIGN, and "integrated" us into an existing UCB
                              028C   658          ;    for the first user of the protocol type.  As a result, we must
                              028C   659          ;    re-lookup the datalink UCB address immediately after the SETMODE,
                              028C   660          ;    and reset our saved value, whether it changed or not.
                              028C   661          ;
                  53   DD  028C   662          PUSHL   R3                        ; Save IRP address
           50   4C A6   3C  028E   663          MOVZWL  DWB$W_DLL_CHAN(R6),R0     ; Get channel number
         00000000'GF   16  0292   664          JSB     G^IOC$VERIFYCHAN          ; Get the CCB address; ignore errors
                  53 8ED0  0298   665          POPL    R3                        ; Restore IRP address
                  61   D0  029B   666          MOVL    CCB$L_UCB(R1),-           ; Save the datalink UCB address
                48 A6      029D   667                  DWB$L_DLL_UCB(R6)
             50   00'   D0  029F   668 40$:     MOVL    S^#SS$_NORMAL,R0          ; Successful
        38 A3   50   B0  02A2   669 90$:     MOVW    R0,IRP$L_IOST1(R3)        ; Store status in IRP
                      05  02A6   670          RSB                               ; Exit with status
                      02A7          .SBTTL  DLE$DEACCESS - Process IO$_DEACCESS request
                      02A7     ;+
                      02A7     ; DLE$DEACCESS - Process IO$_DEACCESS request
                      02A7     ;
                      02A7     ; This routine is called to perform all work needed for the DLE DEACCESS
                      02A7     ; QIO at IPL 0.  If this is a point-to-point circuit, then we must cause
                      02A7     ; the circuit to revert back into its original state.
                      02A7     ;
                      02A7     ; Inputs:
                      02A7     ;
                      02A7     ;       R6 = DWB address
                      02A7     ;       R3 = IRP address
                      02A7     ;
                      02A7     ; Outputs:
                      02A7     ;
                      02A7     ;       R3 = IRP address, 0 if not to be returned to driver yet.
                      02A7     ;       IRP$L_IOST1 = I/O status
                      02A7     ;-
                      02A7    DLE$DEACCESS:
                      02A7            ;
                      02A7            ;    Locate the circuit data structures based on the LPD ID
                      02A7            ;    stored in the DWB at access time.
                      02A7            ;
        54   56  D0   02A7            MOVL    R6,R4                   ; Save DWB address for later
     58  3E  A6  3C   02AA            MOVZWL  DWB$W_PATH(R6),R8       ; Get LPD ID
             41  13   02AE            BEQL    70$                     ; If none, report error
          FD4D'  30   02B0            BSBW    NET$GET_LPD_CRI         ; Get LPD, CRI addresses
        36   50  E9   02B3            BLBC    R0,90$                  ; Exit if error detected
                      02B6            ;
                      02B6            ;    If this is a multiaccess circuit, such as Ethernet,
                      02B6            ;    then skip the circuit transition, since there is no
                      02B6            ;    circuit "mode".
                      02B6            ;
             0A  E1   02B6            BBC     #LPD$V_BC,-             ; Skip if not broadcast
     0D  22  A6       02B8                    LPD$W_STS(R6),20$
                      02BB            $DASSGN_S CHAN=DWB$W_DLL_CHAN(R4) ; Deassign channel to datalink
             24  11   02C6            BRB     90$                     ; Exit with status
                      02C8    20$:
                      02C8            ;
                      02C8            ;    Make sure this user is actually the current "owner"
                      02C8            ;    of the circuit.
                      02C8            ;
                      02C8            $GETFLD cri,l,owpid             ; Get the owner PID
        19   50  E9   02D5            BLBC    R0,70$                  ; If none at all, report an error
     0C  A3  58  D1   02D8            CMPL    R8,IRP$L_PID(R3)        ; Check if this user is owner
             13  12   02DC            BNEQ    70$                     ; If not, return an error
                      02DE            ;
                      02DE            ;    Leave MOP state
                      02DE            ;
           0017  30   02DE            BSBW    LEAVE_MOP_STATE         ; Leave MOP state
                      02E1            ;
                      02E1            ;    Bring the circuit down, which will cause it to attempt
                      02E1            ;    to re-initialize, this time in normal mode (because the
                      02E1            ;    DLE flag is off).
                      02E1            ;
   50  0000'8F   3C   02E1            MOVZWL  #LEV$C_LIN_DOWN,R0      ; Setup DLLTRN event code
          FD17'  30   02E6            BSBW    SET_DLE_EVT             ; Queue the request
        50  00'  D0   02E9            MOVL    S^#SS$_NORMAL,R0        ; Success
```

```
      38 A3    50   B0  02EC   729 90$:    MOVW     R0,IRP$L_IOST1(R3)        ; Store status in IRP
                    05  02F0   730         RSB                                ; Exit with status
                        02F1   731
    50    0000'8F   3C  02F1   732 70$:    MOVZWL   #SS$_FILNOTACC,R0         ; Circuit not accessed
               F4   11  02F6   733         BRB      90$
```

```
                          02F8          .SBTTL  LEAVE_MOP_STATE - Leave MOP state
                          02F8     ;+
                          02F8     ; LEAVE_MOP_STATE - Leave MOP state for an LPD
                          02F8     ;
                          02F8     ; This routine is called to reset LPD fields when leaving MOP state.
                          02F8     ;
                          02F8     ; Inputs:
                          02F8     ;
                          02F8     ;       R10/R11 = CRI pointers
                          02F8     ;       R6 = LPD address
                          02F8     ;
                          02F8     ; Outputs:
                          02F8     ;
                          02F8     ;       None
                          02F8     ;-
                          02F8     LEAVE_MOP_STATE:
                          02F8             ;
                          02F8             ;    Mark the circuit no longer accessed
                          02F8             ;
                          02F8             CLRBIT  #LPD$V_ACCESS,-         ; Mark no longer accessed
                          02F8                     LPD$W_STS(R6)
                          02FD             $CLRFLD cri,l,owpid             ; Clear the owner PID
                          030A             ;
                          030A             ;    If we are just leaving MOP mode, then reset circuit
                          030A             ;    substate and log an event record.
                          030A             ;
              02      E5  030A             BBCC    #LPD$V_DLE,-            ; Clear DLE flag
         21 22 A6         030C                     LPD$W_STS(R6),30$       ; If already cleared, skip following
              0A      90  030F             MOVB    #NMA$C_LINSS_SYN,-      ; Enter "synchronizing" substate
           27 A6          0311                     LPD$B_SUB_STA(R6)
55    00000000'EF     9E  0313             MOVAB   NET$AB_EVT_WQE,R5       ; Get address of common WQE
           20 A6      B0  031A             MOVW    LPD$W_PTH(R6),-         ; Set LPD ID into WQE
           12 A5          031D                     WQE$W_REQIDT(R5)
      0140 8F         B0  031F             MOVW    #EVC$C_DLL_LSC,-        ; "locally initiated state change"
           1C A5          0323                     WQE$W_EVL_CODE(R5)
              04      90  0325             MOVB    #EVC$C_DLC_POLD_MAIN,-  ; Old state = MAINTAINANCE
           1E A5          0327                     WQE$B_EVL_DT1(R5)
              03      90  0329             MOVB    #EVC$C_DLC_POLD_RUNG,-  ; New state = RUNNING
           1F A5          032B                     WQE$B_EVL_DT2(R5)
           FCD0'      30  032D             BSBW    NET$EVT_INTRAW          ; Log the event record
              05          0330     30$:    RSB
```

```
                         0331    777          .SBTTL  DLE$CANCEL - Process DLE cancel request
                         0331    778 ;+
                         0331    779 ; DLE$CANCEL - Process DLE cancel request
                         0331    780 ;
                         0331    781 ; This routine is called to perform all work needed for a cancel of a
                         0331    782 ; DLE "accessed" channel at IPL 0.  Presently, nothing needs to be done
                         0331    783 ; except the datalink cancel I/O already done by the driver.
                         0331    784 ;
                         0331    785 ; Inputs:
                         0331    786 ;
                         0331    787 ;       R3 = IRP address
                         0331    788 ;
                         0331    789 ; Outputs:
                         0331    790 ;
                         0331    791 ;       R3 = IRP address, 0 if not to be returned to driver yet.
                         0331    792 ;       IRP$L_IOST1 = I/O status
                         0331    793 ;-
                         0331    794 DLE$CANCEL:
         50     00'  D0  0331    795          MOVL    S^#SS$_NORMAL,R0           ; Successful
      38 A3     50   B0  0334    796          MOVW    R0,IRP$L_IOST1(R3)         ; Store status in IRP
                     05  0338    797          RSB
```

```
                                  0339         .SBTTL  DLE$BC_UP - Initialize DLE on broadcast circuit
                                  0339 ;+
                                  0339 ; DLE$BC_UP - Initialize DLE on a broadcast circuit which has just come up
                                  0339 ;
                                  0339 ; This routine is called when a broadcast circuit has just come up and
                                  0339 ; entered the "run" state.  It sets up NETACP as the "shared" protocol user
                                  0339 ; of the "load/dump" and "loopback" NI protocols, so that DECnet can
                                  0339 ; receive requests from other nodes on the NI.
                                  0339 ;
                                  0339 ; Inputs:
                                  0339 ;
                                  0339 ;       R11 = CRI CNR address
                                  0339 ;       R10 = CRI CNF address
                                  0339 ;       R7 = ADJ address
                                  0339 ;       R6 = LPD address
                                  0339 ;       R4 = RCB address
                                  0339 ;
                                  0339 ; Outputs:
                                  0339 ;
                                  0339 ;       R0 = Status code
                                  0339 ;
                                  0339 ;       R1 is destroyed.
                                  0339 ;-
                                  0339 DLE$BC_UP::
                  03FC 8F   BB    0339         PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8,R9> ; Save registers
                                  033D ;
                                  033D ;       If service functions are disabled for this circuit, then do
                                  033D ;       not enable "load/dump" or "loopback" protocol types.
                                  033D ;
                                  033D         $GETFLD cri,l,ser                ; Get SERVICE flag
                     64 58   E8   034A         BLBS    R8,90$                   ; Branch if disabled
                                  034D ;
                                  034D ;       Allocate and initialize a new BC context block
                                  034D ;
                     51 2C   3C   034D         MOVZWL  #BC_C_LENGTH,R1          ; Size of structure
              00000000'EF    16   0350         JSB     NET$ALLOCATE             ; Allocate the block
                     5D 50   E9   0356         BLBC    R0,100$                  ; Exit if error detected
                        52   DD   0359         PUSHL   R2                       ; Save address of block
   0C A2   20   00   6E 00   2C   035B         MOVC5   #0,(SP),#0,#BC_C_LENGTH-12,12(R2) ; Zero the block
                     55 8ED0      0362         POPL    R5                       ; Set R5 to block address
               50 24 A5   9E      0365         MOVAB   BC_Q_UNSOL_MSGS(R5),R0   ; Get address of listhead
                  60 50   D0      0369         MOVL    R0,(R0)                  ; Init listhead
                  60 80   DE      036C         MOVAL   (R0)+,(R0)
               50 14 A5   9E      036F         MOVAB   BC_Q_PND_RCV(R5),R0      ; Get address of listhead
                  60 50   D0      0373         MOVL    R0,(R0)                  ; Init listhead
                  60 80   DE      0376         MOVAL   (R0)+,(R0)
               50 1C A5   9E      0379         MOVAB   BC_Q_CUR_RCV(R5),R0      ; Get address of listhead
                  60 50   D0      037D         MOVL    R0,(R0)                  ; Init listhead
                  60 80   DE      0380         MOVAL   (R0)+,(R0)
         0E A5  20 A6   B0        0383         MOVW    LPD$W_PTH(R6),BC_W_LPD(R5) ; Save LPD of associated circuit
         0000000C'FF  65  0E      0388         INSQUE  (R5),@BC_QUEUE+4         ; Insert block into queue
                                  038F ;
                                  038F ;       Initialize ourselves as the "default user" of the "load/dump"
                                  038F ;       protocol type.
                                  038F ;
            53  10 A5   3E        038F         MOVAW   BC_W_LD_CHAN(R5),R3      ; Point to word to receive channel #
     54   00000053'EF   9E        0393         MOVAB   LD_SETMODE,R4            ; Point to descriptor of SETMODE buffer
```

```
            00A5   30  039A               BSBW    INIT_UNSOL_CHAN            ; Initialize channel
           16 50   F9  039D               BLBC    R0,100$                    ; Exit if error detected
                       03A0           ;
                       03A0           ;   Initialize ourselves as the "default user" of the "loopback"
                       03A0           ;   protocol type.
                       03A0           ;
       53   12 A5  3E  03A0               MOVAW   BC_W_LP_CHAN(R5),R3        ; Point to word to receive channel #
  54   0000009D'EF 9E  03A4               MOVAB   LP_SETMODE,R4              ; Point to descriptor of SETMODE buffer
            0094   30  03AB               BSBW    INIT_UNSOL_CHAN            ; Initialize channel
           05 50   E9  03AE               BLBC    R0,100$                    ; Branch if error detected
         03FC 8F   BA  03B1      90$:     POPR    #^M<R2,R3,R4,R5,R6,R7,R8,R9> ; Restore registers
              05       03B5               RSB                                ; Exit with status
                       03B6
                       03B6  ;
                       03B6  ; An error occurred trying to setup the circuit for service functions.
                       03B6  ; Log an error, and bring down the circuit.
                       03B6  ;
                       03B6
  55   00000000'EF 9E  03B6      100$:    MOVAB   NET$AB_EVT_WQE,R5          ; Get address of common WQE
              07   B0  03BD               MOVW    #EVC$C_NMA_ABS,-           ; "aborted service request"
           1C A5       03BF                       WQE$W_EVL_CODE(R5)
              04   90  03C1               MOVB    #EVC$C_NMA_PRSN_LOE,-      ; Reason = "line open error"
           1E A5       03C3                       WQE$B_EVL_DT1(R5)
           FC38'   30  03C5               BSBW    NET$EVT_INTRAW             ; Log the event record
     50    0000'8F 3C  03C8               MOVZWL  #LEV$C_CIN_DOWN,R0         ; Setup "circuit down" event
           FC30'   30  03CD               BSBW    SET_DLE_EVT                ; Queue event to DLLTRN
              DF   11  03D0               BRB     90$                        ; Exit
```

```
                              03D2            .SBTTL  DLE$BC_DOWN - Cleanup DLE on broadcast circuit
                              03D2  ;+
                              03D2  ; DLE$BC_DOWN - Cleanup DLE on broadcast circuit
                              03D2  ;
                              03D2  ; This routine is called when a broadcast circuit leaves the "run" state.
                              03D2  ; We must deallocate any BC context blocks if they were associated with this
                              03D2  ; circuit.
                              03D2  ;
                              03D2  ; Inputs:
                              03D2  ;
                              03D2  ;       R6 = LPD address
                              03D2  ;
                              03D2  ; Outputs:
                              03D2  ;
                              03D2  ;       None
                              03D2  ;-
                              03D2  DLE$BC_DOWN::
                    3C   BB   03D2          PUSHR   #^M<R2,R3,R4,R5>          ; Save registers
                              03D4          ;
                              03D4          ;    Locate the BC block associated with this circuit.
                              03D4          ;
      51   00000008'EF   9E   03D4          MOVAB   BC_QUEUE,R1               ; Get address of BC queue
               55   51   D0   03DB          MOVL    R1,R5                     ; Setup for loop
               55   65   D0   03DE  10$:    MOVL    (R5),R5                   ; Skip to next block in queue
               51   55   D1   03E1          CMPL    R5,R1                     ; End of list?
                    59   13   03E4          BEQL    90$                       ; If not found, skip it
            0E  A5       B1   03E6          CMPW    BC_W_LPD(R5),-            ; Does the LPD ID match?
            20  A6            03E9                  LPD$W_PTH(R6)
                    F1   12   03EB          BNEQ    10$                       ; If not, keep looking
               55   65   0F   03ED          REMQUE  (R5),R5                   ; Remove BC from list
                              03F0          ;
                              03F0          ;    For any non-zero channels, deassign them
                              03F0          ;
      50    10  A5       3C   03F0          MOVZWL  BC_W_LD_CHAN(R5),R0       ; Get "load/dump" channel
                    0A   13   03F4          BEQL    20$                       ; If nonzero,
                              03F6          $DASSGN_S CHAN=R0                 ; Deassign it
      50    12  A5       3C   0400  20$:    MOVZWL  BC_W_LP_CHAN(R5),R0       ; Get "loopback" channel
                    0A   13   0404          BEQL    30$                       ; If nonzero,
                              0406          $DASSGN_S CHAN=R0                 ; Deassign it
                              0410  30$:    ;
                              0410          ;    Deallocate all unsolicited messages still waiting for
                              0410          ;    the process to deal with them.
                              0410          ;
      50    24  B5       0F   0410  40$:    REMQUE  @BC_Q_UNSOL_MSGS(R5),R0   ; Get next unsolicited message
                    08   1D   0414          BVS     45$                       ; Branch if none left in queue
           00000000'EF   16   0416          JSB     NET$DEALLOCATE            ; Deallocate the block
                    F2   11   041C          BRB     40$                       ; Empty the entire queue
                              041E  45$:    ;
                              041E          ;    Deallocate all receive IOWQEs waiting to be issued to
                              041E          ;    the NI driver.
                              041E          ;
      50    14  B5       0F   041E  60$:    REMQUE  @BC_Q_PND_RCV(R5),R0      ; Get next waiting receive IOWQE
                    08   1D   0422          BVS     65$                       ; Branch if none left in queue
           00000000'EF   16   0424          JSB     NET$DEALLOCATE            ; Deallocate the block
                    F2   11   042A          BRB     60$                       ; Empty the entire queue
                              042C  65$:    ;
                              042C          ;    Deallocate the BC context block
```

```
                                 042C   941            ;
                                 042C   942            SETBIT  #BC_V_DELETE,BC_B_FLAGS(R5) ; Mark block for deletion
                 0C A5    95     0431   943            TSTB    BC_B_REFCNT(R5)              ; Are there still receives outstanding?
                    09    12     0434   944            BNEQ    90$                          ; If so, wait for them to complete
                                 0436   945                                                 ; before deallocating the block
              50    55    D0     0436   946            MOVL    R5,R0                        ; Set the block address
            00000000'EF   16     0439   947            JSB     NET$DEALLOCATE               ; Deallocate it
                    3C    BA     043F   948 90$:       POPR    #^M<R2,R3,R4,R5>             ; Restore registers
                          05     0441   949            RSB
```

```
                           0442          .SBTTL  INIT_UNSOL_CHAN - Initialize channel for unsolicited msgs
                           0442   ;+
                           0442   ; INIT_UNSOL_CHAN - Initialize channel for unsolicited messages for a protocol
                           0442   ;
                           0442   ; This routine is called to assign a new datalink channel, setup the channel
                           0442   ; to be the "default user" of the protocol, so that messages not directly
                           0442   ; intended for any other "limited users" of the protocol come to us, and then
                           0442   ; issue an asynchronous recieve on the channel.
                           0442   ;
                           0442   ; Inputs:
                           0442   ;
                           0442   ;        R10/R11 = CRI pointers
                           0442   ;        R3 = Address of word to store channel number
                           0442   ;        R4 = Address of SETMODE P2 buffer
                           0442   ;        R5 = Address of BC context block
                           0442   ;
                           0442   ; Outputs:
                           0442   ;
                           0442   ;        R0 = Status code
                           0442   ;-
                           0442   INIT_UNSOL_CHAN:
      50    0000'8F    3C  0442          MOVZWL  #SS$_NOSUCHDEV,R0          ; Setup default error status
                           0447          $GETFLD cri,s,vmsnam              ; Get datalink device name
          6C 50        E9  0454          BLBC    R0,90$                     ; Branch if error detected
        7E    57       7D  0457          MOVQ    R7,-(SP)                   ; Push descriptor on stack
        50    5E       D0  045A          MOVL    SP,R0                      ; Get address of descriptor
                           045D          $ASSIGN_S DEVNAM=(R0),-            ; Assign channel to NI driver
                           045D                  CHAN=(R3)
        5E    08       C0  046A          ADDL    #8,SP                      ; Pop descriptor off stack
          53 50        E9  046D          BLBC    R0,90$                     ; Branch if error detected
                           0470          ;
                           0470          ;    Issue a SETMODE request to the NI driver to establish the
                           0470          ;    channels as accessing the protocol type as "default user".
                           0470          ;
                           0470          $QIOW_S FUNC=#IO$_SETMODE!IO$M_CTRL!IO$M_STARTUP,-
                           0470                  CHAN=(R3),-
                           0470                  EFN=#NET$C_EFN_WAIT,-
                           0470                  IOSB=IOSB,-
                           0470                  P2=R4
          2F 50        E9  0491          BLBC    R0,90$                     ; Branch if error detected
   50  00000010'EF     3C  0494          MOVZWL  IOSB,R0                    ; Get final I/O status
          25 50        E9  049B          BLBC    R0,90$                     ; Branch if error detected
                           049E          ;
                           049E          ;    Allocate and initialize an IOWQE to to be used to receive
                           049E          ;    unsolicited messages for this protocol.
                           049E          ;
      51    05FE 8F    3C  049E          MOVZWL  #IOWQE_C_LENGTH-WQE$C_LENGTH,R1 ; Get additional storage size
          50    03     D0  04A3          MOVL    #WQE$C_SUB_AST,R0          ; Indicate WQE sub-type
             FB57'     30  04A6          BSBW    WQE$ALLOCATE               ; Allocate a WQE - always succeeds
       2C A2    63     B0  04A9          MOVW    (R3),IOWQE_W_CHAN(R2)      ; Store channel to datalink
       34 A2    55     D0  04AD          MOVL    R5,IOWQE_L_BC(R2)          ; Store backpointer to BC block
          0E A5        B0  04B1          MOVW    BC_W_LPD(R5),-             ; Use LPD ID as REQIDT
          12 A2            04B4                  WQE$Q_REQIDT(R2)
    18 B5    62        0E  04B6          INSQUE  (R2),@BC_Q_PND_RCV+4(R5); Insert on pending receive queue
                           04BA          ;
                           04BA          ;    Issue asynchronous read on the channel, so that we are
                           04BA          ;    notified when someone sends us an unsolicited message.
```

```
                        04BA   1008         ;
  51    0E A5    3C     04BA   1009         MOVZWL  BC_W_LPD(R5),R1         ; Get LPD index
           04    10     04BE   1010         BSBB    ISSUE_NI_READ           ; Issue read request
     50    00'   D0     04C0   1011         MOVL    S^#SS$_NORMAL,R0        ; Success
                 05     04C3   1012 90$:    RSB                             ; Exit with status
```

```
NE
Sy

$$
$$
AB
AB
AB
AB
AB
AC
AC
AC
AC
AC
AC
AT
BC
BC
BC
BC
BC
BC
BC
BC
BC
BC
BC
BC
BC
BC
BI
CC
CN
CN
CN
CN
CN
CN
CN
CX
CX
CX
CX
CX
DD
DL
DL
DL
DL
DL
DL
DL
DL
DL
DL
```

```
                                04C4          .SBTTL  ISSUE_NI_READ - Issue read request to NI driver
                                04C4   ;+
                                04C4   ; ISSUE_NI_READ - Issue read request to NI driver
                                04C4   ;
                                04C4   ; This routine is called to issue the read request, and return as soon
                                04C4   ; as the request has been queued.  All read requests are automatically
                                04C4   ; delayed by 1 second, so that if there is an abnormal node continuously
                                04C4   ; sending messages, we won't get swamped (the NI driver will drop them
                                04C4   ; for us).  The delay doesn't affect normal reception, because the NI
                                04C4   ; driver buffers any incoming messages for us, up to a limit.
                                04C4   ;
                                04C4   ; Inputs:
                                04C4   ;
                                04C4   ;       R1 = LPD ID for circuit
                                04C4   ;
                                04C4   ; Outputs:
                                04C4   ;
                                04C4   ;       None
                                04C4   ;
                                04C4   ;       R0-R1 are destroyed.
                                04C4   ;-
                                04C4   ISSUE_NI_READ:
                        0C  BB  04C4          PUSHR   #^M<R2,R3>              ; Save registers
              51    51  10  78  04C6          ASHL    #16,R1,R1               ; Shift LPD ID into upper word
              51  0401  8F  B0  04CA          MOVW    #<<WQE$C_QUAL_DLE>@8>!- ; Overlay QUAL and EVT fields
                                04CF                  TID_C_READSUP,R1
                52    E4'AF  9E  04CF          MOVAB   B^50$,R2                ; Set address of action routine
  53   00000000 00989680 8F  7D  04D3          MOVQ    #1*10*1000*1000,R3      ; Wait 1 second
                     FB1F'  30  04DE          BSBW    WQE$RESET_TIM           ; Wait for timer to fire
                        0C  BA  04E1          POPR    #^M<R2,R3>              ; Restore registers
                        05      04E3          RSB
                                04E4   ;
                                04E4   ; Call here when timer fires
                                04E4   ;
              58    12 A5   3C  04E4   50$:   MOVZWL  WQE$W_REQIDT(R5),R8     ; Get LPD ID
                 50    55   D0  04E8          MOVL    R5,R0                   ; Get timer WQE address
                     FB12'  30  04EB          BSBW    WQE$DEALLOCATE          ; Deallocate timer WQE
                                04EE          ;
                                04EE          ;       Locate the BC block associated with this circuit.  When found,
                                04EE          ;       if there are any IOWQEs (receives) waiting to be issued to the
                                04EE          ;       NI driver, issue them now.
                                04EE          ;
          54    00000008'EF  9E  04EE          MOVAB   BC_QUEUE,R4             ; Get address of BC queue
                    55  54  D0  04F5          MOVL    R4,R5                   ; Setup for loop
                    55  65  D0  04F8   5$:    MOVL    (R5),R5                 ; Skip to next block in queue
                    54  55  D1  04FB          CMPL    R5,R4                   ; End of list?
                        4D  13  04FE          BEQL    90$                     ; If not found, skip it
              58    0E A5   B1  0500          CMPW    BC_W_LPD(R5),R8         ; Does the LPD ID match?
                        F2  12  0504          BNEQ    5$                      ; If not, keep looking
              52    14 B5   0F  0506   10$:   REMQUE  @BC_Q_PND_RCV(R5),R2    ; Get any receives waiting to be issued
                        EC  1D  050A          BVS     5$                      ; If none, keep looking
           20 B5    62  0E  050C          INSQUE  (R2),@BC_Q_CUR_RCV+4(R5); Insert on outstanding receive queue
                    0C A5   96  0510          INCB    BC_B_REFCNT(R5)         ; Increment reference count
              50    38 A2   9E  0513          MOVAB   IOWQE_G_NIHDR(R2),R0    ; Get address of NI header buffer
                                0517          $QIO_S  FUNC=#IO$_READVBLK,-    ; Wait for a message to come in
                                0517                  CHAN=IOWQE_W_CHAN(R2),-
                                0517                  EFN=#NET$C_EFN_ASYN,-
```

```
                       0517                          IOSB=IOWQE_Q_IOSB(R2),-
                       0517                          ASTADR=B^RCV_DLE_MSG_AST,-
                       0517                          ASTPRM=R2,-
                       0517                          P1=IOWQE_G_MSG(R2),-      ; Address of receive buffer
                       0517                          P2=#1500,-                ; Length of receive buffer
                       0517                          P5=R0                     ; Address of buffer to receive NI header
          C5 50    E8  053E                BLBS      R0,10$                    ; Branch if ok
       24 A2  50   3C  0541                MOVZWL    R0,IOWQE_Q_IOSB(R2)       ; Store QIO status in IOSB
              52   DD  0545                PUSHL     R2                        ; IOWQE address
       4E'AF  01   FB  0547                CALLS     #1,B^RCV_DLE_MSG_AST      ; Call AST routine
          B9  11       054B                BRB       10$                       ; Keep scanning
              05       054D       90$:     RSB
                       054E
                       054E       ;
                       054E       ; Receive AST
                       054E       ;
                       054E
                       054E       RCV_DLE_MSG_AST:
                 0000  054E                .WORD     0
                       0550
      50   04 AC   D0  0550                MOVL      4(AP),R0                  ; Get WQE address
                       0554                ;
                       0554                ;    Remove from outstanding receive queue
                       0554                ;
        50   60    0F  0554                REMQUE    (R0),R0                   ; Remove from queue
                       0557                ;
                       0557                ;    Queue a work queue entry to process the I/O completion
                       0557                ;
          60'AF    9E  0557                MOVAB     B^RCV_DLE_MSG,-           ; Set address of work routine
          0C A0        055A                          WQE$L_ACTION(R0)
           FAA1'   30  055C                BSBW      WQE$INSQUE                ; Insert onto work queue
                   04  055F                RET
```

```
                              0560          .SBTTL  RCV_DLE_MSG - Receive unsolicited DLE message
                              0560    ;+
                              0560    ; RCV_DLE_MSG - Receive unsolicited DLE message
                              0560    ;
                              0560    ; This routine is called when a receive completes on one of the DLE "shared"
                              0560    ; channels.  This means that an unsolicited message has come in which could
                              0560    ; not be associated with any existing protocol user.  Our action is to start
                              0560    ; up a MOM process to handle the DLE session.
                              0560    ;
                              0560    ; Inputs:
                              0560    ;
                              0560    ;       R5 = IOWQE address
                              0560    ;
                              0560    ; Outputs:
                              0560    ;
                              0560    ;       None
                              0560    ;-
                              0560    RCV_DLE_MSG:
        54    34 A5    D0     0560          MOVL    IOWQE_L_BC(R5),R4       ; Get BC address
              0C A4    97     0564          DECB    BC_B_REFCNT(R4)         ; Decrement outstanding I/O count
                              0567          ;
                              0567          ;   Locate the CRI associated with this circuit
                              0567          ;
        58    0E A4    3C     0567          MOVZWL  BC_W_LPD(R4),R8         ; Get LPD ID
               FA92'   30     056B          BSBW    NET$GET_LPD_CRI         ; Get LPD, CRI addresses
              2C 50    E9     056E          BLBC    R0,5$                   ; Exit if error detected
                              0571          ;
                              0571          ;   If the BC is marked for rundown, then this I/O completion
                              0571          ;   should be ignored, and the BC deallocated if possible.
                              0571          ;
   10 0B A4      00    E1     0571          BBC     #BC_V_DELETE,BC_B_FLAGS(R4),4$ ; If BC marked for rundown,
           0C A4       95     0576          TSTB    BC_B_REFCNT(R4)         ; Any more receives still outstanding?
                 22    12     0579          BNEQ    5$                      ; If so, don't deallocate BC yet
         50      54    D0     057B          MOVL    R4,R0                   ; Set address of BC
      00000000'EF      16     057E          JSB     NET$DEALLOCATE          ; Deallocate BC
                 17    11     0584          BRB     5$                      ; and deallocate IOWQE as well
                              0586    4$:   ;
                              0586          ;   If I/O status was not successful, then stop doing any I/O
                              0586          ;   on this channel (assume it is in the process of running down).
                              0586          ;
        1D 24 A5       E8     0586          BLBS    IOWQE_Q_IOSB(R5),10$    ; If I/O failure,
                 07    B0     058A          MOVW    #EVC$C_NMA_ABS,-        ; "Aborted service request"
              1C A5           058C                  WQE$W_EVL_CODE(R5)
                 01    90     058E          MOVB    #EVC$C_NMA_PRSN_ERR,-   ; "Receive error"
              1E A5           0590                  WQE$B_EVL_DT1(R5)
               FA6B'   30     0592          BSBW    NET$EVT_INTRAW          ; Log the event record
        50   0000'8F   3C     0595          MOVZWL  #LEV$C_CIN_DOWN,R0      ; Setup "circuit down" event
               FA63'   30     059A          BSBW    SET_DLE_EVT             ; Queue event to DLLTRN
         50      55    D0     059D    5$:   MOVL    R5,R0                   ; Get IOWQE address
      00000000'EF      16     05A0          JSB     NET$DEALLOCATE          ; Deallocate it
                       05     05A6          RSB
                              05A7    10$:  ;
                              05A7          ;   If there is already an unsolicited message received from
                              05A7          ;   the remote node waiting for the MOM process to startup,
                              05A7          ;   then drop the message on the floor - don't startup a
                              05A7          ;   redundant MOM process for the same node.
                              05A7          ;
      51    24 A4    9E  05A7         MOVAB   BC_Q_UNSOL_MSGS(R4),R1  ; Get address of unsolicited msg queue
         52    51    D0  05AB         MOVL    R1,R2                   ; Setup for loop
         52    62    D0  05AE  15$:   MOVL    (R2),R2                 ; Skip to next msg in list
         51    52    D1  05B1         CMPL    R2,R1                   ; End of list?
               0E    13  05B4         BEQL    20$                     ; If so, then skip it
               06    BB  05B6         PUSHR   #^M<R1,R2>              ; Save registers
               0E    29  05B8         CMPC    #NIHDRSIZ,-             ; Does the NI header match?
            38 A2        05BA                 IOWQE_G_NIHDR(R2),-
            38 A5        05BC                 IOWQE_G_NIHDR(R5)
               06    BA  05BE         POPR    #^M<R1,R2>              ; Restore registers
               EC    12  05C0         BNEQ    15$                     ; If it doesn't match, keep looking
               06    11  05C2         BRB     30$                     ; If match found, drop msg on floor
                         05C4  20$:   ;
                         05C4         ;       Startup a process to deal with the message
                         05C4         ;
             00AE    30  05C4         BSBW    STARTUP_MOM             ; Start MOM process
                         05C7         ;
                         05C7         ;       If the process could not be created, re-issue the read
                         05C7         ;       request using the same buffer.
                         05C7         ;
            0C 50    E8  05C7         BLBS    R0,40$                  ; Branch if successful
      18 B4    65    0E  05CA  30$:   INSQUE  (R5),@BC_Q_PND_RCV+4(R4); Insert on pending receive queue
      51    0E A4    3C  05CE         MOVZWL  BC_W_LPD(R4),R1         ; Get LPD ID
             FEEF    30  05D2         BSBW    ISSUE_NI_READ           ; Re-issue read request
               05        05D5         RSB
                         05D6         ;
                         05D6         ;       Save PID of MOM process just started in unsolicited message
                         05D6         ;       context block.  From now on, this message is "tagged" for
                         05D6         ;       that process: If the process comes in with an ACCESS function,
                         05D6         ;       we give it the message; if the process dies, we deallocate the
                         05D6         ;       message.
                         05D6         ;
      30 A5    51    D0  05D6  40$:   MOVL    R1,IOWQE_L_PID(R5)      ; Save PID of associated MOM process
                         05DA         ;
                         05DA         ;       Insert the message on the queue waiting for the process to
                         05DA         ;       get started.
                         05DA         ;
      28 B4    65    0E  05DA         INSQUE  (R5),@BC_Q_UNSOL_MSGS+4(R4)     ; Insert at end of queue
                         05DE         ;
                         05DE         ;       Re-issue another receive request for this protocol type
                         05DE         ;
   51    05FE 8F     3C  05DE         MOVZWL  #IOWQE_C_LENGTH-WQE$C_LENGTH,R1 ; Get additional storage size
         50    03    D0  05E3         MOVL    #WQE$C_SUB_AST,R0       ; Indicate WQE sub-type
            FA17'    30  05E6         BSBW    WQE$ALLOCATE            ; Allocate a WQE - always succeeds
         2C A5       B0  05E9         MOVW    IOWQE_W_CHAN(R5),-      ; Copy channel to datalink
         2C A2           05EC                 IOWQE_W_CHAN(R2)
         34 A5       D0  05EE         MOVL    IOWQE_L_BC(R5),-        ; Copy backpointer to BC block
         34 A2           05F1                 IOWQE_L_BC(R2)
         12 A5       B0  05F3         MOVW    WQE$W_REQIDT(R5),-      ; Use the same REQIDT
         12 A2           05F6                 WQE$W_REQIDT(R2)
      18 B4    62    0E  05F8         INSQUE  (R2),@BC_Q_PND_RCV+4(R4); Insert on pending receive queue
      51    0E A4    3C  05FC         MOVZWL  BC_W_LPD(R4),R1         ; Get LPD ID
             FEC1    30  0600         BSBW    ISSUE_NI_READ           ; Issue another read request
               05        0603  90$:   RSB
```

```
                                  0604          .SBTTL  DLE$MOP_REQUEST - Partner has requested MOP mode
                                  0604  ;+
                                  0604  ; DLE$MOP_REQUEST - The circuit partner has requested MOP mode
                                  0604  ;
                                  0604  ; This routine is called when the datalink has received a MOP message
                                  0604  ; from the partner node on a point-to-point datalink.
                                  0604  ;
                                  0604  ; Inputs:
                                  0604  ;
                                  0604  ;       R10/R11 = CRI pointers
                                  0604  ;       R6 = LPD address
                                  0604  ;
                                  0604  ; Outputs:
                                  0604  ;
                                  0604  ;       None
                                  0604  ;
                                  0604  ;       R0-R3,R8-R9 are destroyed.
                                  0604  ;-
                                  0604  DLE$MOP_REQUEST::
                 00F0 8F    BB    0604          PUSHR   #^M<R4,R5,R6,R7>        ; Save registers
                      02    E2    0608          BBSS    #LPD$V_DLE,-            ; Mark circuit in MOP mode
                 1D 22 A6         060A                  LPD$W_STS(R6),10$       ; If already marked, skip logging event
                                  060D          ;
                                  060D          ;       Log an event indicating the circuit has gone into MOP mode
                                  060D          ;
       55    00000000'EF    9E    060D          MOVAB   NET$AB_EVT_WQE,R5       ; Get address of common WQE
                   20 A6    B0    0614          MOVW    LPD$W_PTH(R6),-         ; Set LPD ID into WQE
                   12 A5          0617                  WQE$W_REQIDT(R5)
                 0141 8F    B0    0619          MOVW    #EVC$C_DLL_RSC,-        ; "remotely initiated state change"
                   1C A5          061D                  WQE$W_EVL_CODE(R5)
                      03    90    061F          MOVB    #EVC$C_DLL_POLD_RUNG,-  ; Old state = RUNNING
                   1E A5          0621                  WQE$B_EVL_DT1(R5)
                      04    90    0623          MOVB    #EVC$C_DLL_POLD_MAIN,-  ; New state = MAINTAINANCE
                   1F A5          0625                  WQE$B_EVL_DT2(R5)
                   F9D6'    30    0627          BSBW    NET$EVT_INTRAW          ; Log the event record
                                  062A          ;
                                  062A          ;       If circuit is already accessed, then ignore MOP notification
                                  062A          ;
                      03    E0    062A  10$:    BBS     #LPD$V_ACCESS,-         ; Branch if circuit accessed
                 2A 22 A6         062C                  LPD$W_STS(R6),40$
                                  062F          ;
                                  062F          ;       If service functions are disabled for this circuit, then
                                  062F          ;       ignore MOP request, and recycle circuit.
                                  062F          ;
                                  062F          $GETFLD cri,l,ser               ; Get SERVICE flag
                   24 58    E8    063C          BLBS    R8,50$                  ; Branch if disabled
                                  063F          ;
                                  063F          ;       Set the circuit substate to "auto-service"
                                  063F          ;
                      06    90    063F          MOVB    #NMA$C_LINSS_ASE,-      ; Set circuit substate
                   27 A6          0641                  LPD$B_SUB_STA(R6)
                                  0643          ;
                                  0643          ;       Startup a process to deal with the message
                                  0643          ;
                    002F    30    0643          BSBW    STARTUP_MOM             ; Start MOM process
                   1A 50    E9    0646          BLBC    R0,50$                  ; Branch if unsuccessful
                                  0649          ;
```

```
                          0649          ;   Save PID of MOM process just started in CRI block
                          0649          ;
        58    51    D0    0649          MOVL    R1,R8                       ; Setup PID of created process
                          064C          $PUTFLD cri,l,owpid                 ; Set DLE owner of circuit
                          0659          ;
                          0659          ;   We can respond to the MOP request.  Recycle the circuit
                          0659          ;   and force it to come up in "MOP" state (because of the
                          0659          ;   DLE flag).
                          0659          ;
    50  0000'8F     3C    0659     40$: MOVZWL  #LEV$C_LIN_DOWN,R0          ; Setup "line down" event
           F99F'    30    065E          BSBW    SET_DLE_EVT                 ; Queue the event
              0D    11    0661          BRB     90$
                          0663
                          0663          ;
                          0663          ;   We cannot respond to the MOP request.  Recycle the circuit
                          0663          ;   and force it to come back in regular mode.
                          0663          ;
                          0663     50$: CLRBIT  #LPD$V_DLE,LPD$W_STS(R6) ; Mark circuit in "normal" mode
    50  0000'8F     3C    0668          MOVZWL  #LEV$C_LIN_DOWN,R0          ; Setup "line down" event
           F990'    30    066D          BSBW    SET_DLE_EVT                 ; Queue the event
         00F0 8F    BA    0670     90$: POPR    #^M<R4,R5,R6,R7>            ; Restore registers
                    05    0674          RSB
```

```
                                0675           .SBTTL  STARTUP_MOM - Start MOM process
                                0675   ;+
                                0675   ; STARTUP_MOM - Start MOM process for auto-service
                                0675   ;
                                0675   ; This routine is called to start the MOM process.
                                0675   ;
                                0675   ; Inputs:
                                0675   ;
                                0675   ;       R10/R11 = CRI pointers
                                0675   ;
                                0675   ; Outputs:
                                0675   ;
                                0675   ;       R0 = Status code
                                0675   ;       R1 = IPID of process, if successful
                                0675   ;
                                0675   ;       R2-R3,R7-R9 are destroyed.
                                0675   ;-
                                0675   STARTUP_MOM:
                   30     BB    0675           PUSHR   #^M<R4,R5>                      ; Save registers
                                0677           $GETFLD cri,s,nam                       ; Get circuit name
                                0684           ;
                                0684           ;   Repeatly try to startup MOM, and if it fails due to "duplicate
                                0684           ;   process name", then try again with another process name until
                                0684           ;   it suceeds.
                                0684           ;
              59   01     D0    0684           MOVL    #1,R9                           ; Start with postfix #1
              5E   0C     C2    0687   10$:    SUBL    #12,SP                          ; Allocate prcnam buffer on stack
                   5E     DD    068A           PUSHL   SP                              ; Construct descriptor of buffer
                   0C     DD    068C           PUSHL   #12
    51      00000005'EF   9E    068E           MOVAB   MOM_PRCNAM,R1                   ; Get address of FAO string
              50   81     9A    0695           MOVZBL  (R1)+,R0                        ; Construct descriptor of FAO string
              7E   50     7D    0698           MOVQ    R0,-(SP)                        ; Push FAO descriptor onto stack
              50   5E     D0    069B           MOVL    SP,R0                           ; Get stack address
                                069E           $FAO_S  CTRSTR=(R0),-                   ; Construct process name
                                069E                   OUTBUF=8(R0),-
                                069E                   OUTLEN=8(R0),-
                                069E                   P1=R7,-                         ; Length of circuit name
                                069E                   P2=R8,-                         ; Address of circuit name
                                069E                   P3=R9                           ; Process number
              5E   08     C0    06B3           ADDL    #8,SP                           ; Pop FAO string descriptor
              54   8E     7D    06B6           MOVQ    (SP)+,R4                        ; R4/R5 = descriptor of process name
              52   57     7D    06B9           MOVQ    R7,R2                           ; Pass circuit name as SYS$NET
            0180   8F     BB    06BC           PUSHR   #^M<R7,R8>                      ; Save circuit name
    58      00000000'EF   9E    06C0           MOVAB   MOM_OBJ_NAM,R8                  ; Point to ASCIC MOM object name
              57   88     9A    06C7           MOVZBL  (R8)+,R7                        ; Construct descriptor of name
                F933'     30    06CA           BSBW    NET$STARTUP_OBJ_NAM             ; Startup the object
            0180   8F     BA    06CD           POPR    #^M<R7,R8>                      ; Restore circuit name
              5E   0C     C0    06D1           ADDL    #12,SP                          ; Pop process name buffer
         0000'8F   50     B1    06D4           CMPW    R0,#SS$_DUPLNAM                 ; Process name already exist?
                   08     12    06D9           BNEQ    20$                             ; If so,
  FFA6 59    01    0A     F1    06DB           ACBL    #MAX_MOM_PROC,#1,R9,10$         ; Increment number and try again
                   1A     11    06E1           BRB     90$                             ; Exit with error, but don't log
                                06E3                                                   ; any error - too many MOMs already
                                06E3   20$:    ;
                                06E3           ;   If the process could not be created, log an event record.
                                06E3           ;
               17 50      E8    06E3           BLBS    R0,90$                          ; Branch if successful
```

```
                    50   DD  06E6  1353        PUSHL   R0                          ; Save status
     55   00000000'EF    9E  06E8  1354        MOVAB   NET$AB_EVT_WQE,R5           ; Get address of common WQE
                    07   B0  06EF  1355        MOVW    #EVC$C_NMA_ABS,-            ; "aborted service request"
                 1C A5       06F1  1356                WQE$W_EVL_CODE(R5)
                    04   90  06F3  1357        MOVB    #EVC$C_NMA_PRSN_LOE,-       ; Reason = "line open error"
                 1E A5       06F5  1358                WQE$B_EVL_DT1(R5)
                 F906'   30  06F7  1359        BSBW    NET$EVT_INTRAW              ; Log the event record
                 50    8ED0  06FA  1360        POPL    R0                          ; Restore status
                 30      BA  06FD  1361 90$:   POPR    #^M<R4,R5>                  ; Restore registers
                         05  06FF  1362        RSB
```

```
                                0700          .SBTTL  ATTACH_UNSOL_MSG - Attach unsolicited message
                                0700  ;+
                                0700  ; ATTACH_UNSOL_MSG - Attach unsolicited message to newly accessed DWB
                                0700  ;
                                0700  ; This routine is called to search the unsolicited message queue, and
                                0700  ; if one is found for this DLE user, to insert the message onto it's
                                0700  ; private receive queue.
                                0700  ;
                                0700  ; Inputs:
                                0700  ;
                                0700  ;       R3 = IO$_ACCESS IRP address
                                0700  ;       R4 = DWB adress
                                0700  ;
                                0700  ; Outputs:
                                0700  ;
                                0700  ;       IRP$L_EXTEND(R3) = Address of CXB containing unsolicited message
                                0700  ;                          (or zero if no message found)
                                0700  ;
                                0700  ;               CXB$W_LENGTH    = Message length in bytes (not incl. NI header)
                                0700  ;               CXB$C_HEADER    = 14-byte NI datalink header
                                0700  ;               CXB$C_HEADER+14 = Message
                                0700  ;
                                0700  ;       R0-R1 are destroyed.
                                0700  ;-
                                0700  ATTACH_UNSOL_MSG:
                  55     DD     0700          PUSHL   R5                      ; Save registers
               54 A3     D4     0702          CLRL    IRP$L_EXTEND(R3)        ; Preset no CXB address
                                0705          ;
                                0705          ;    Locate the BC block associated with this circuit.
                                0705          ;
  51    00000008'EF      9E     0705          MOVAB   BC_QUEUE,R1             ; Get address of BC queue
            55    51     D0     070C          MOVL    R1,R5                   ; Setup for loop
            55    65     D0     070F  5$:     MOVL    (R5),R5                 ; Skip to next block in queue
            51    55     D1     0712          CMPL    R5,R1                   ; End of list?
                  5A     13     0715          BEQL    90$                     ; If not found, skip it
               0E A5     B1     0717          CMPW    BC_W_LPD(R5),-          ; Does the LPD ID match?
               3E A4            071A                  DWB$W_PATH(R4)
                  F1     12     071C          BNEQ    5$                      ; If not, keep looking
                                071E          ;
                                071E          ;    Search for unsolicited message which was "tagged" for
                                071E          ;    this process.
                                071E          ;
         51    24 A5     9E     071E          MOVAB   BC_Q_UNSOL_MSGS(R5),R1  ; Get address of unsolicited msg queue
            55    51     D0     0722          MOVL    R1,R5                   ; Setup for loop
            55    65     D0     0725  10$:    MOVL    (R5),R5                 ; Skip to next msg in list
            51    55     D1     0728          CMPL    R5,R1                   ; End of list?
                  44     13     072B          BEQL    90$                     ; If so, then skip it
               30 A5     D1     072D          CMPL    IOWQE_L_PID(R5),-       ; Does the IPID match?
               0C A3            0730                  IRP$L_PID(R3)
                  F1     12     0732          BNEQ    10$                     ; If not, keep looking
                                0734          ;
                                0734          ;    Allocate a CXB from non-paged pool, store the message into
                                0734          ;    the block, and insert it into the DWB receive queue.
                                0734          ;
         51    26 A5     3C     0734          MOVZWL  IOWQE_W_MSGLEN(R5),R1   ; Get size of message
  51    0000005A 8F      C0     0738          ADDL    #CXB$C_OVERHEAD+NIHDRSIZ,R1 ; Compute size of CXB
        00000000'EF      16     073F          JSB     NET$ALONPAGED           ; Allocate from non-paged pool
```

```
          29 50    E9 0745          BLBC    R0,90$                    ; If insufficient memory, skip it
    08 A2    51    B0 0748          MOVW    R1,CXB$W_SIZE(R2)         ; Set size of structure
    0A A2    1B    90 074C          MOVB    #DYN$C_CXB,CXB$B_TYPE(R2) ; Set type of structure
    62    56 A2    9E 0750          MOVAB   CXB$C_HEADER+-            ; Set data area address in CXB
                      0754                  NIHDRSIZ(R2),(R2)
          26 A5    B0 0754          MOVW    IOWQE_W_MSGLEN(R5),-      ; Save message size in CXB
          0C A2       0757                  CXB$W_LENGTH(R2)
    54 A3    52    D0 0759          MOVL    R2,IRP$L_EXTEND(R3)       ; Save address of CXB
             3C    BB 075D          PUSHR   #^M<R2,R3,R4,R5>          ; Save registers
             0E    28 075F          MOVC    #NIHDRSIZ,-               ; Copy NI datalink header
          38 A5       0761                  IOWQE_G_NIHDR(R5),-
          48 A2       0763                  CXB$C_HEADER(R2)
    55    0C AE    D0 0765          MOVL    3*4(SP),R5                ; Recover IOWQE address
          26 A5    28 0769          MOVC    IOWQE_W_MSGLEN(R5),-      ; Copy message
          46 A5       076C                  IOWQE_G_MSG(R5),-
             63       076E                  (R3)
             3C    BA 076F          POPR    #^M<R2,R3,R4,R5>          ; Restore registers
             55  8ED0 0771     90$: POPL    R5                        ; Restore registers
                   05 0774          RSB
```

```
                              0775                    .SBTTL  DLE$PRC_EXIT - Handle MOM process termination
                              0775          ;+
                              0775          ; DLE$PRC_EXIT - Handle MOM process termination
                              0775          ;
                              0775          ; This routine is called whenever any process "owned" by NETACP terminates.
                              0775          ; We must check if we have any unsolicited MOP messages intended for the
                              0775          ; terminated process, and if so, clean them up.
                              0775          ;
                              0775          ; Inputs:
                              0775          ;
                              0775          ;       R8 = IPID of terminated process
                              0775          ;
                              0775          ; Outputs:
                              0775          ;
                              0775          ;       None
                              0775          ;-
                              0775          DLE$PRC_EXIT::
                              0775                  ;
                              0775                  ;   Scan all broadcast circuits
                              0775                  ;
51     00000008'EF       9E   0775                  MOVAB   BC_QUEUE,R1             ; Get address of BC queue
             55   51     D0   077C                  MOVL    R1,R5                   ; Setup for loop
             55   65     D0   077F          5$:     MOVL    (R5),R5                 ; Skip to next block in queue
             51   55     D1   0782                  CMPL    R5,R1                   ; End of list?
                  25     13   0785                  BEQL    20$                     ; If not found, skip it
                              0787                  ;
                              0787                  ;   Deallocate any messages which are intended for this process
                              0787                  ;
        52     24 A5     9E   0787                  MOVAB   BC_Q_UNSOL_MSGS(R5),R2  ; Get address of unsolicited msg queue
             53   52     D0   078B                  MOVL    R2,R3                   ; Setup for loop
             53   63     D0   078E          10$:    MOVL    (R3),R3                 ; Skip to next msg in list
             52   53     D1   0791          15$:    CMPL    R3,R2                   ; End of list?
                  E9     13   0794                  BEQL    5$                      ; If so, then continue to next circuit
        58     30 A3     D1   0796                  CMPL    IOWQE_L_PID(R3),R8      ; Does the IPID match?
                  F2     12   079A                  BNEQ    10$                     ; If not, keep looking
                  63     DD   079C                  PUSHL   (R3)                    ; Save pointer to next block in list
             50   63     0F   079E                  REMQUE  (R3),R0                 ; Remove it from the queue
       00000000'EF       16   07A1                  JSB     NET$DEALLOCATE          ; Deallocate the block
             53 8ED0          07A7                  POPL    R3                      ; Set R3 to next block in list
                  E5     11   07AA                  BRB     15$                     ; Keep looking for more
                              07AC          20$:    ;
                              07AC                  ;   If any circuits are in MOP state waiting for the MOM
                              07AC                  ;   process to issue its initial ACCESS, then reset them
                              07AC                  ;   back into normal state.  We recognize this condition
                              07AC                  ;   if the OWPID field is still set to the PID, meaning
                              07AC                  ;   that the process must never have accessed the DLE
                              07AC                  ;   channel (or else we would have cleared it on DEACCESS).
                              07AC                  ;
5B     00000000'EF       D0   07AC                  MOVL    NET$GL_CNR_CRI,R11      ; Point to CRI database
                  5A     D4   07B3                  CLRL    R10                     ; Start at beginning
                              07B5          25$:    $SEARCH eql,cri,l,owpid         ; Search for circuits
           13 50         E9   07C4                  BLBC    R0,30$                  ; Branch if none found
             F836'       30   07C7                  BSBW    NET$LOCATE_LPD          ; Locate associated LPD
           E8 50         E9   07CA                  BLBC    R0,25$                  ; If error detected, skip it
             FB28        30   07CD                  BSBW    LEAVE_MOP_STATE         ; Return circuit to normal mode
     50    0000'8F       3C   07D0                  MOVZWL  #LEV$C_LIN_DOWN,R0      ; Setup "line down" event
             F828'       30   07D5                  BSBW    SET_DLC_EVT             ; Queue the event
```

```
                               DB  11  07D8  1498         BRB     25$                              ; Keep looping
                                   05  07DA  1499 30$:    RSB
                                       07DB  1500
                                       07DB  1501
                                       07DB  1502         .END
```

```
NETDLE                                   - NETACP DLE processing          D  2       16-SEP-1984 01:24:27  VAX/VMS Macro V04-00        Page  38
Symbol table                                                                          5-SEP-1984 02:19:17  [NETACP.SRC]NETDLE.MAR;1        (19)

$$T1                          = 00000000                        DWB$B_SUBSTA                  = 00000046
$$T2                          = 00000006                        DWB$L_DLL_UCB                 = 00000048
ABD$C_FIB                     = 00000001                        DWB$V_BC                      = 00000003
ABD$C_LENGTH                  = 00000008                        DWB$W_DLL_CHAN                = 0000004C
ABD$C_NAME                    = 00000002                        DWB$W_FLAGS                   = 0000000E
ABD$W_COUNT                   = 00000002                        DWB$W_PATH                    = 0000003E
ABD$W_TEXT                    = 00000000                        DYN$C_CXB                     = 00000018
ACP$C_STA_F                   = 00000004                        EVC$C_DLL_LSC                 = 00000140
ACP$C_STA_H                   = 00000005                        EVC$C_DLL_POLD_MAIN           = 00000004
ACP$C_STA_I                   = 00000000                        EVC$C_DLL_POLD_RUNG           = 00000003
ACP$C_STA_N                   = 00000001                        EVC$C_DLL_RSC                 = 00000141
ACP$C_STA_R                   = 00000002                        EVC$C_NMA_ABS                 = 00000007
ACP$C_STA_S                   = 00000003                        EVC$C_NMA_PRSN_ERR            = 00000001
ATTACH_UNSOL_MSG                00000700 R        04            EVC$C_NMA_PRSN_LOE            = 00000004
BC_ACCESS                       000001C0 R        04            EXE$INSIOQ                      ********    X   04
BC_B_FLAGS                      0000000B  G                     INIT_UNSOL_CHAN                 00000442 R        04
BC_B_REFCNT                     0000000C  G                     IO$M_CTRL                     = 00000200
BC_B_TYPE                       0000000A  G                     IO$M_STARTUP                  = 00000040
BC_C_LENGTH                     0000002C  G                     IO$_ACCESS                    = 00000032
BC_L_FLINK                      00000000  G                     IO$_ACPCONTROL                = 00000038
BC_M_DELETE                   = 00000001  G                     IO$_DEACCESS                  = 00000034
BC_QUEUE                        00000008 R        02            IO$_READVBLK                  = 00000031
BC_Q_CUR_RCV                    0000001C  G                     IO$_SETMODE                   = 00000023
BC_Q_PND_RCV                    00000014  G                     IOC$VERIFYCHAN                  ********    X   04
BC_Q_UNSOL_MSGS                 00000024  G                     IOSB                            00000010 R        02
BC_V_DELETE                   = 00000000  G                     IOWQE_C_LENGTH                  00000622  G
BC_W_LD_CHAN                    00000010  G                     IOWQE_G_MSG                     00000046  G
BC_W_LPD                        0000000E  G                     IOWQE_G_NIHDR                   00000038  G
BC_W_LP_CHAN                    00000012  G                     IOWQE_L_BC                      00000034  G
BC_W_SIZE                       00000008  G                     IOWQE_L_PID                     00000030  G
BIT...                        = 00000001                        IOWQE_Q_IOSB                    00000024  G
CCB$L_UCB                     = 00000000                        IOWQE_W_CHAN                    0000002C  G
CNF$CLR_FIELD                   ********    X     04            IOWQE_W_MSGLEN                = 00000026
CNF$GET_FIELD                   ********    X     04            IRP$L_DIAGBUF                 = 0000004C
CNF$KEY_SEARCH                  ********    X     04            IRP$L_EXTEND                  = 00000054
CNF$PUT_FIELD                   ********    X     04            IRP$L_IOST1                   = 00000038
CNF$_ADVANCE                  = 00000000                        IRP$L_IOST2                   = 0000003C
CNF$_QUIT                     = 00000002                        IRP$L_PID                     = 0000000C
CNF$_TAKE_CURR                = 00000003                        IRP$L_SVAPTE                  = 0000002C
CNF$_TAKE_PREV                = 00000001                        IRP$L_UCB                     = 0000001C
CXB$B_TYPE                    = 0000000A                        IRP$L_WIND                    = 00000018
CXB$C_HEADER                  = 00000048                        IRP$S_FCODE                   = 00000006
CXB$C_OVERHEAD                = 0000004C                        IRP$V_COMPLX                  = 00000003
CXB$W_LENGTH                  = 0000000C                        IRP$V_FCODE                   = 00000000
CXB$W_SIZE                    = 00000008                        IRP$W_FUNC                    = 00000020
DDT$L_UNSOLINT                = 00000004                        IRP$W_STS                     = 0000002A
DLE$ACCESS                      00000075 R        04            ISSUE_NI_READ                   000004C4 R        04
DLE$BC_DOWN                     000003D2 RG       04            LD_PARAMS                       00000011 R        03
DLE$BC_UP                       00000339 RG       04            LD_SETMODE                      00000053 R        03
DLE$CANCEL                      00000331 R        04            LEAVE_MOP_STATE                 000002F8 R        04
DLE$DEACCESS                    000002A7 R        04            LEV$C_DLE_ACC                   ********    X   04
DLE$DISPATCH                    00000000 RG       04            LEV$C_LIN_DOWN                  ********    X   04
DLE$LPD_STATUS                  00000157 RG       04            LPD$B_SUB_STA                 = 00000027
DLE$MOP_REQUEST                 00000604 RG       04            LPD$L_UCB                     = 00000010
DLE$PRC_EXIT                    00000775 RG       04            LPD$V_ACCESS                  = 00000003
DLE$SETMODE                     0000021F R        04            LPD$V_BC                      = 0000000A
DLE_ACC                         00000000 R        02            LPD$V_DLE                     = 00000002
```

```
LPD$V_RUN                        = 00000004                      NMA$C_PCLI_BUS                   = 00000AF1
LPD$V_X25                        = 00000007                      NMA$C_PCLI_CRC                   = 00000B1C
LPD$W_CHAN                       = 00000014                      NMA$C_PCLI_DCH                   = 00000B1B
LPD$W_PTH                        = 00000020                      NMA$C_PCLI_MCA                   = 00000B0F
LPD$W_STS                        = 00000022                      NMA$C_PCLI_MLT                   = 00000B19
LP_PARAMS                          0000005B R       03           NMA$C_PCLI_PAD                   = 00000B1A
LP_SETMODE                         0000009D R       03           NMA$C_PCLI_PRM                   = 00000B18
MAX_MOM_PROC                     = 0000000A                      NMA$C_PCLI_PTY                   = 00000B0E
MOM_OBJ_NAM                        00000000 R       03           NMA$C_STATE_OFF                  = 00000001
MOM_PRCNAM                         00000005 R       03           NMA$C_STATE_ON                   = 00000000
NET$AB_EVT_WQE                     ********      X  04           NSP$C_EXT_LNK                    = 0000001E
NET$ALLOCATE                       ********      X  04           NSP$C_MAXHDR                     = 00000009
NET$ALONPAGED                      ********      X  04           RCV_DLE_MSG                        00000560 R       04
NET$C_ACT_TIMER                  = 0000001E                      RCV_DLE_MSG_AST                    0000054E R       04
NET$C_EFN_ASYN                   = 00000002                      SET_DLL_EVT                        ********      X  04
NET$C_EFN_WAIT                   = 00000001                      SIZ...                           = 00000001
NET$C_IPL                        = 00000008                      SS$_DEVALLOC                       ********      X  04
NET$C_MAXACCFLD                  = 00000027                      SS$_DEVINACT                       ********      X  04
NET$C_MAXLINNAM                  = 0000000F                      SS$_DUPLNAM                        ********      X  04
NET$C_MAXLNK                     = 000003FF                      SS$_FILNOTACC                      ********      X  04
NET$C_MAXNODNAM                  = 00000006                      SS$_ILLIOFUNC                      ********      X  04
NET$C_MAXOBJNAM                  = 0000000C                      SS$_IVMODE                         ********      X  04
NET$C_MAX_AREAS                  = 0000003F                      SS$_NORMAL                         ********      X  04
NET$C_MAX_LINES                  = 00000040                      SS$_NOSUCHDEV                      ********      X  04
NET$C_MAX_NCB                    = 0000006E                      STARTUP_MOM                        00000675 R       04
NET$C_MAX_NODES                  = 000003FF                      SYS$ASSIGN                         ********     GX  04
NET$C_MAX_OBJ                    = 000000FF                      SYS$DASSGN                         ********     GX  04
NET$C_MAX_WQE                    = 00000014                      SYS$FAO                            ********      X  04
NET$C_MINBUFSIZ                  = 000000C0                      SYS$QIO                            ********     GX  04
NET$C_TID_ACT                    = 00000003                      SYS$QIOW                           ********     GX  04
NET$C_TID_RUS                    = 00000001                      TID_C_READSUP                    = 00000001
NET$C_TID_XRT                    = 00000002                      TR$C_MAXHDR                      = 0000001C
NET$C_TRCTL_CEL                  = 00000002                      TR$C_NI_ALLEND1                  = 040000AB
NET$C_TRCTL_OVR                  = 00000005                      TR$C_NI_ALLEND2                  = 00000000
NET$C_UTLBUFSIZ                  = 00001000                      TR$C_NI_ALLROU1                  = 030000AB
NET$DEALLOCATE                     ********      X  04           TR$C_NI_ALLROU2                  = 00000000
NET$EVT_INTRAW                     ********      X  04           TR$C_NI_PREFIX                   = 000400AA
NET$FIND_LPD                       ********      X  04           TR$C_NI_PROT                     = 00000360
NET$GET_LPD_CRI                    ********      X  04           TR$C_PRI_ECL                     = 0000001F
NET$GL_CNR_CRI                     ********      X  04           TR$C_PRI_RTHRU                   = 0000001F
NET$GL_DLE_UCB                     ********      X  04           UCB$C_DDT                        = 00000088
NET$LOCATE_LPD                     ********      X  04           WQE$ALLOCATE                       ********      X  04
NET$M_MAXLNKMSK                  = 000003FF                      WQE$B_EVL_DT1                    = 0000001E
NET$STARTUP_OBJ_NAM                ********      X  04           WQE$B_EVL_DT2                    = 0000001F
NFB$C_CRI_NAM                    = 04020041                      WQE$C_LENGTH                     = 00000024
NFB$C_CRI_OWPID                  = 04010010                      WQE$C_QUAL_DLE                   = 00000004
NFB$C_CRI_SER                    = 04000002                      WQE$C_SUB_AST                    = 00000003
NFB$C_CRI_STA                    = 04010013                      WQE$DEALLOCATE                     ********      X  04
NFB$C_CRI_VMSNAM                 = 04020042                      WQE$INSQUE                         ********      X  04
NFB$C_OP_EQL                     = 00000000                      WQE$L_ACTION                     = 0000000C
NIHDRSIZ                         = 0000000E                      WQE$RESET_TIM                      ********      X  04
NMA$C_ACC_SHR                    = 00000001                      WQE$W_EVL_CODE                   = 0000001C
NMA$C_LINMC_SET                  = 00000001                      WQE$W_REQIDT                     = 00000012
NMA$C_LINSS_ASE                  = 00000006                      _$$_                             = 00000000
NMA$C_LINSS_SYN                  = 0000000A
NMA$C_PCLI_ACC                   = 00000B1E
NMA$C_PCLI_BFN                   = 00000451
```

```
                                              +----------------+
                                              ! Psect synopsis !
                                              +----------------+

PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
.  ABS  .                       00000000  (    0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                           00000622  ( 1570.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE
NET_IMPURE                      00000018  (   24.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR NOEXE   RD     WRT NOVEC LONG
NET_PURE                        000000A5  (  165.)  03 (  3.)  NOPIC   USR   CON   REL   LCL NOSHR NOEXE   RD   NOWRT NOVEC LONG
NET_CODE                        000007DB  ( 2011.)  04 (  4.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD   NOWRT NOVEC BYTE

                                         +------------------------+
                                         ! Performance indicators !
                                         +------------------------+

Phase                    Page faults    CPU Time       Elapsed Time
-----                    -----------    --------       ------------
Initialization                   27     00:00:00.11    00:00:00.57
Command processing              152     00:00:01.10    00:00:04.42
Pass 1                          831     00:00:32.24    00:00:43.49
Symbol table sort                 0     00:00:04.69    00:00:05.06
Pass 2                          376     00:00:06.40    00:00:08.42
Symbol table output              31     00:00:00.21    00:00:00.22
Psect synopsis output             4     00:00:00.03    00:00:00.03
Cross-reference output            0     00:00:00.00    00:00:00.00
Assembler run totals           1423     00:00:44.80    00:01:02.23

The working set limit was 2000 pages.
178619 bytes (349 pages) of virtual memory were used to buffer the intermediate code.
There were 180 pages of symbol table space allocated to hold 3209 non-local and 73 local symbols.
1502 source lines were read in Pass 1, producing 26 object records in Pass 2.
58 pages of virtual memory were used to define 53 macros.

                                        +--------------------------+
                                        ! Macro library statistics !
                                        +--------------------------+

Macro library name                        Macros defined
------------------                        --------------
_$255$DUA28:[SHRLIB]NMALIBRY.MLB;1                     1
_$255$DUA28:[SHRLIB]EVCDEF.MLB;1                       1
_$255$DUA28:[NETACP.OBJ]NETDRV.MLB;1                   0
_$255$DUA28:[NETACP.OBJ]NET.MLB;1                     13
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                        10
_$255$DUA28:[SYSLIB]STARLET.MLB;2                     17
TOTALS (all libraries)                                42

3530 GETS were required to define 42 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:NETDLE/OBJ=OBJ$:NETDLE MSRC$:NETDLE/UPDATE=(ENH$:NETDLE)+EXECML$/LIB+LIB$:NET/LIB+LIB$:NETDRV/LIB+SHRLIB$:EVCDEF/LIB+
```

0275 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

NETDLE
LIS

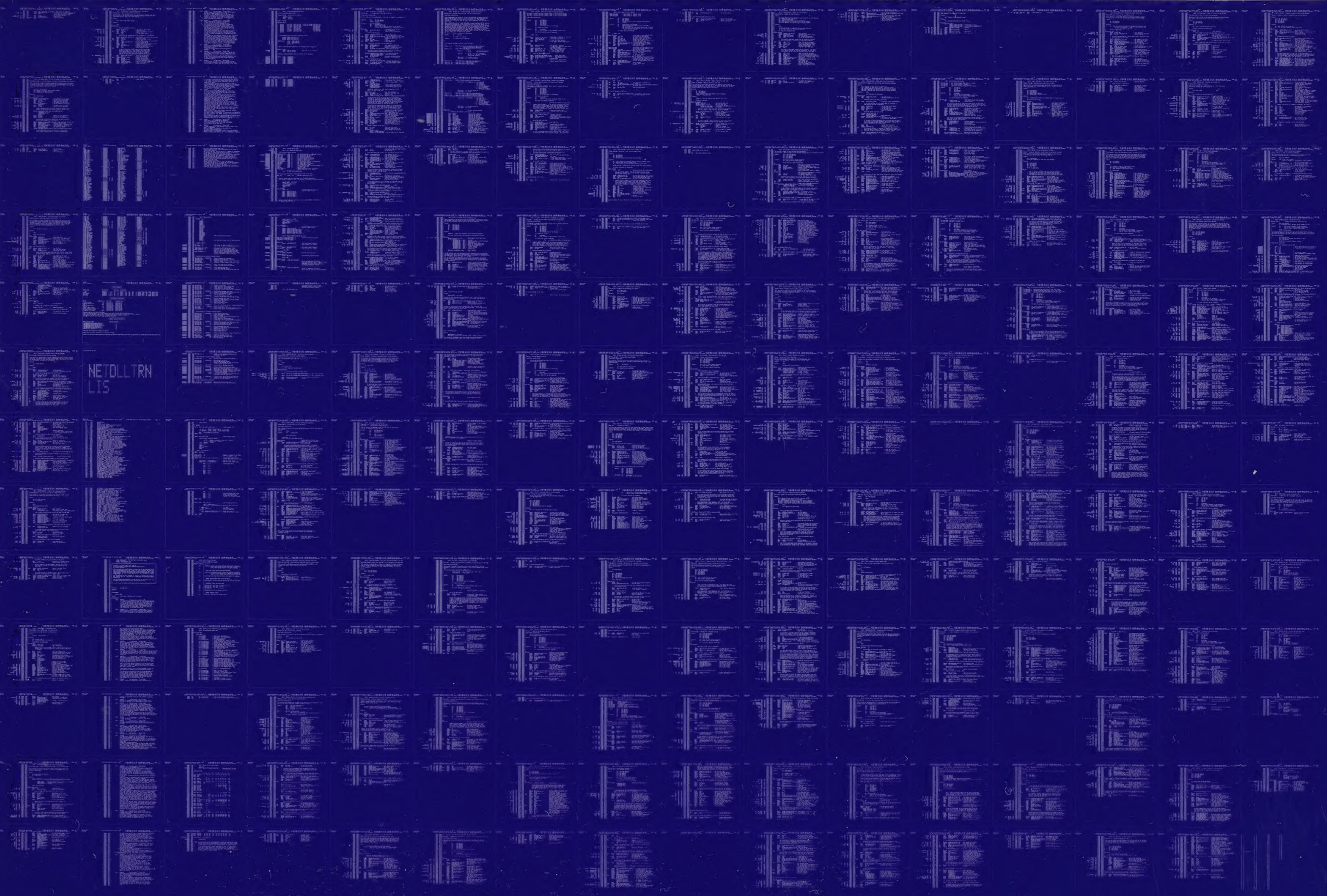
NETCONFIG
LIS

NETCONECT
LIS

NETCTLALL
LIS

0276 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



NETDLLTRN
LIS